

August 1977

**Printer Control with
the UPI-41^{T.M.}**

Lionel Smith
Microcomputer Applications

Related Intel Publications

“UPI-41TM User's Manual”

The material in this Application Note is for informational purposes only and is subject to change without notice. Intel Corporation has made an effort to verify that the material in this document is correct. However, Intel Corporation does not assume any responsibility for errors that may appear in this document.

The following are trademarks of Intel Corporation and may be used only to describe Intel Products:

ICE
INSITE
INTEL
INTELLEC
LIBRARY MANAGER

MCS
MEGACHASSIS
MICROAMP
PROMPT
UPI

Printer Control with the UPI-41^{T.M.}

Contents

INTRODUCTION	1
THE LRC PRINTER	1
INTERFACE SIGNALS	2
TIMING	6
SOFTWARE	6
DETAILS OF THE BUFFER MANAGER	7
PRINTER SERVICE ROUTINES	9
CONCLUSION	13
APPENDIX	13

INTRODUCTION

The UPI-41 is a low-cost, single-chip microcomputer designed to be used as a universal peripheral interface device in a microcomputer system. The device is based on a completely self-contained 8-bit microcomputer with program memory, data memory, CPU, I/O, event timer, and clock oscillator, in a single 40-pin package. A bus interface is included which enables the UPI-41 to be used as a peripheral controller in MCS-48, MCS-80, MCS-85 and other 8-bit microcomputer families. The device is designed for keyboard scanning, printer control, display multiplexing and similar applications which involve interfacing peripheral devices to microcomputer systems.

The UPI-41 is fabricated with N-channel MOS technology and requires only a single 5-volt supply for operation. It has 1K words of program memory and 64 words of data memory on-chip. Both ROM (8041) and EPROM (8741) versions are available and the two are completely pin compatible. The instruction set of the UPI-41 is almost identical to that of the MCS-48. A single byte data register on the UPI-41 interfaces directly to an 8-bit master processor bus to handle asynchronous data transfer to and from the master system. A separate 4-bit register is used to indicate the status of data transfer. Two 8-bit TTL-compatible I/O ports plus two single-bit test inputs are available. I/O can be expanded further by using the 8243 I/O expander device. A separate register in the UPI-41 is used as an event counter or interval timer.

Because it is a complete microcomputer, the UPI-41 provides more power and flexibility than conventional LSI interface devices. For instance, the UPI-41 can be programmed as a peripheral interface for any of the low-cost drum or dot matrix printers currently on the market. In addition to controlling the printer, the UPI-41 can handle zero suppression, limit-checking, formatting and other computations, thereby unburdening the master processor. This type of distributed intelligence, made possible by the UPI-41, greatly enhances overall system capability while reducing cost and development time.

This application note describes how the UPI-41 can be used to implement an interface to a matrix printer. The printer chosen is fairly typical of a large class of printers which minimize total system cost by reducing the mechanical content at the expense of more sophisticated electronic requirements. The UPI-41, with its high degree of capabil-

ity, is ideal for this type of application. It is suggested that the reader not already familiar with the UPI-41 read the "Intel UPI-41 User's Manual" before proceeding in this document.

THE LRC PRINTER

The LRC Model 7040 printer is a matrix printer manufactured by LRC Inc. of Riverton, Wyoming. Capable of printing up to 40 columns of alphanumeric information, this printer is mechanically simple and should be ideal for a variety of applications such as point of sale terminals and data logging. While this note concentrates on the Model 7040 printer, the techniques discussed should be applicable to a variety of similar printers which are currently available.

The printer (Figure 1) consists of four major sub-assemblies, the frame, the print head, the main drive, and the paper handling components. The frame is an aluminum extrusion which provides a suitable base for mounting the various components of the printer. The print head consists of seven solenoids which each drive stiff wires to impact the paper through the inked ribbon. At the solenoid end of the print head these wires are arranged in a circular fashion. Where these wires impact the printer, however, the wires are arranged in a vertical column. To see how this arrangement can be used to print alphanumeric characters refer to Figure 2. The figure shows a 5 X 7 matrix of "dots". The columns are labeled C1 through C5; the rows are labeled as Row 1 through Row 7. Each row corresponds to one of the solenoid-driven wires. The entire print head assembly is moved left to right across the paper so that at T_1 it is over C1, at T_2 it is over C2, and so on. If the correct solenoids are activated at each of these times (T_1 – T_5) then a character can be formed. Figure 2 shows the character "A" formed. At T_1 solenoids one through five were active, at T_2 solenoids four and six were active, and so on until the complete character was formed. The complete character is formed by choosing the correct pattern of active solenoids for each of five instants in time.

The print head is moved across the paper by the main drive. The main drive consists of a 24-pole synchronous motor which drives a rotating plastic drum. The drum has a spiral groove molded into it. A pin attached to the print head rests in this groove so that as the drum rotates at a constant speed the print head is driven back and forth across the paper. Printing is accomplished by controlling

the activation of the solenoids as the print head is driven from left to right across the paper. When the end of the print area occurs the spiral groove reverses the direction of the head motion. As the left-hand edge of the paper is reached a cam attached to the drum activates the HOME micro-switch and the groove again reverses the motion of the head. When the print head is again over the print area and travelling in the left to right direction the microswitch is deactivated. The printer controller uses the trailing edge of the signal generated by the microswitch to initiate the printing of a new line of information.

Paper feed is accomplished by a second synchronous motor which can be activated to feed paper through the mechanism. A switch is provided which is activated while the actual line feed is occurring. The control logic can use the trailing

edge of the signal generated by this switch to turn off the line feed motor. A version of the printer with automatic line feed is available.

INTERFACE SIGNALS

The interface signals to the printer consists of a pair of wires for each solenoid, a pair of wires for each motor (main drive and line feed), a pair of wires returning the state of the HOME micro-switch, and a pair of wires returning the state of the LINEFEED microswitch.

The solenoids must be driven from a 40 ± 4 volt source. The peak current is approximately 3.6A, the average current is approximately 0.5A. A circuit providing the required drive is shown in Figure 3. The output stage, consisting of the 2N6045 Darlington transistor, the 1N4002 catching diode, and the 20-ohm damping resistor, is the

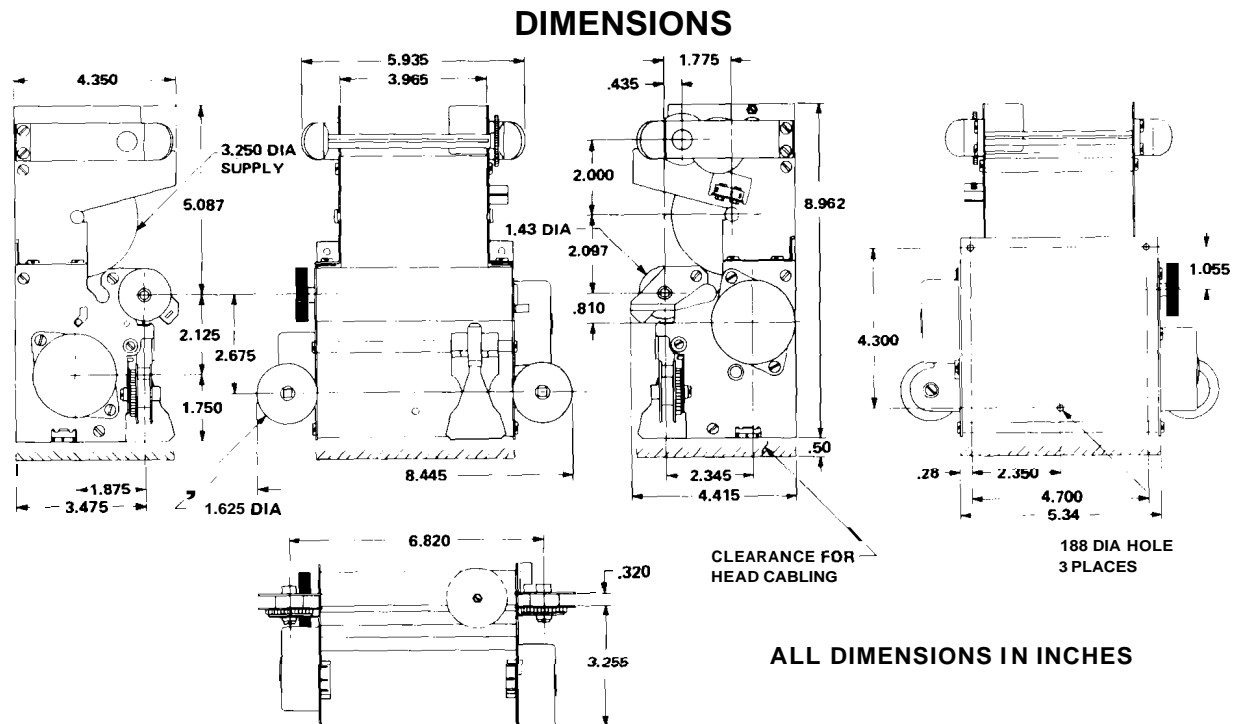


Figure 1. LRC Model 7040 Printer

one suggested by the manufacturer of the printer. The input stage is a discrete implementation of a DTL gate. Note that the base-emitter junction of the 2N6045 will protect the 2N2222A transistor from over-voltage on its collector. This circuit has several features which are important to the printer interface:

1. All solenoid power (including the power used to drive the base of the power transistor) is derived from the 40-volt supply.
2. Disconnecting the drivers from the UPI-41 or the loss of the 5-volt supply to the UPI-41 will result in the solenoids being turned off.

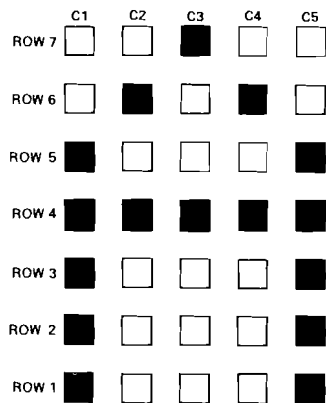


Figure 2. 5 x 7 Dot Matrix

The first feature of the drivers will minimize the impact of the printer and its interface on the 5-volt supply of the system. The second feature prevents the activation of the solenoids erroneously during power on/off cycles or during system checkout. This is an important point since the solenoids will be damaged if left activated continuously. (During the debug of the design described in this note fuses were added to the solenoid drivers to protect them from mishap.)

The two motors can each be driven as shown in Figure 4. The Monsanto MCS-6200 is an optically-coupled TRIAC which is ideal for driving the small synchronous motors in the printer. Coupled with a buffer this part provides a simple means of controlling the motor without sacrificing the isolation required for safe and reliable operation.

Figure 5 shows a UPI-41 used as an interface between an Intel[®] 8085 and an LRC Model 7040 printer. The drivers which have already been described have been used to interface the TTL outputs of the 8741 to the levels required by the printer. The two contact closure outputs from the printer (PAPERFEED and HOME) have been filtered and applied to the TEST0 and TEST1 inputs of the UPI-41. Bit 5 of output port 2 has been designated as an interrupt pin which will be used to request service from the 8085.

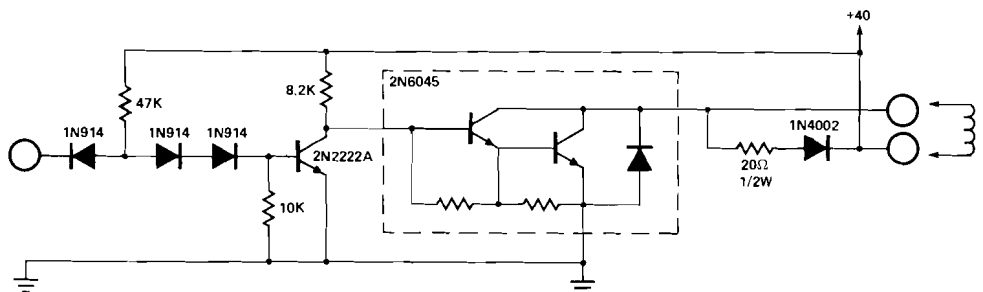


Figure 3. Solenoid Driver

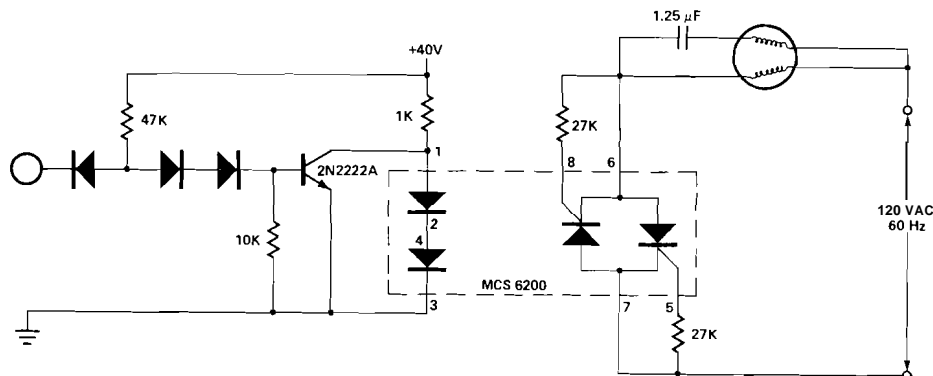


Figure 4. Motor Driver

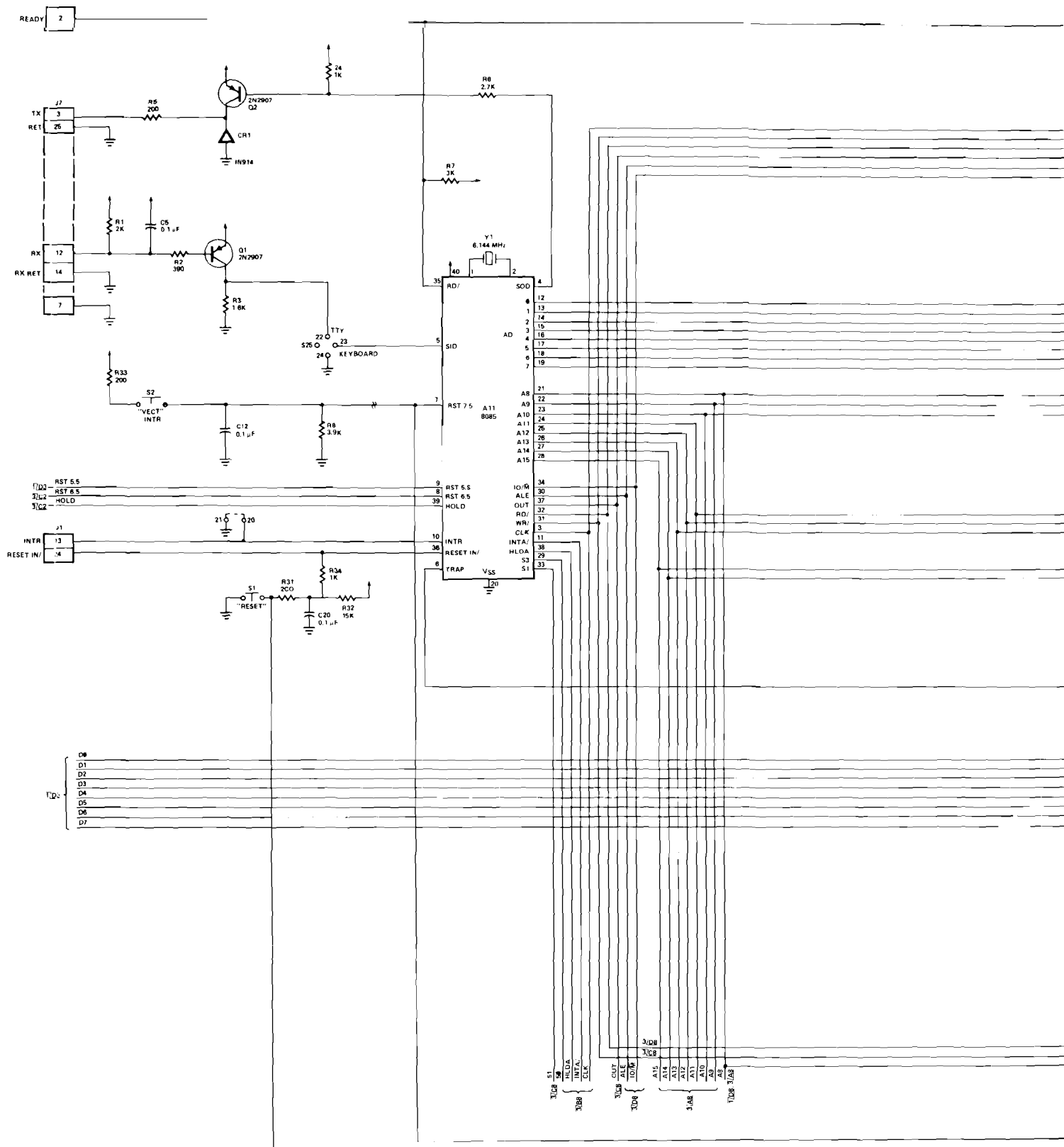
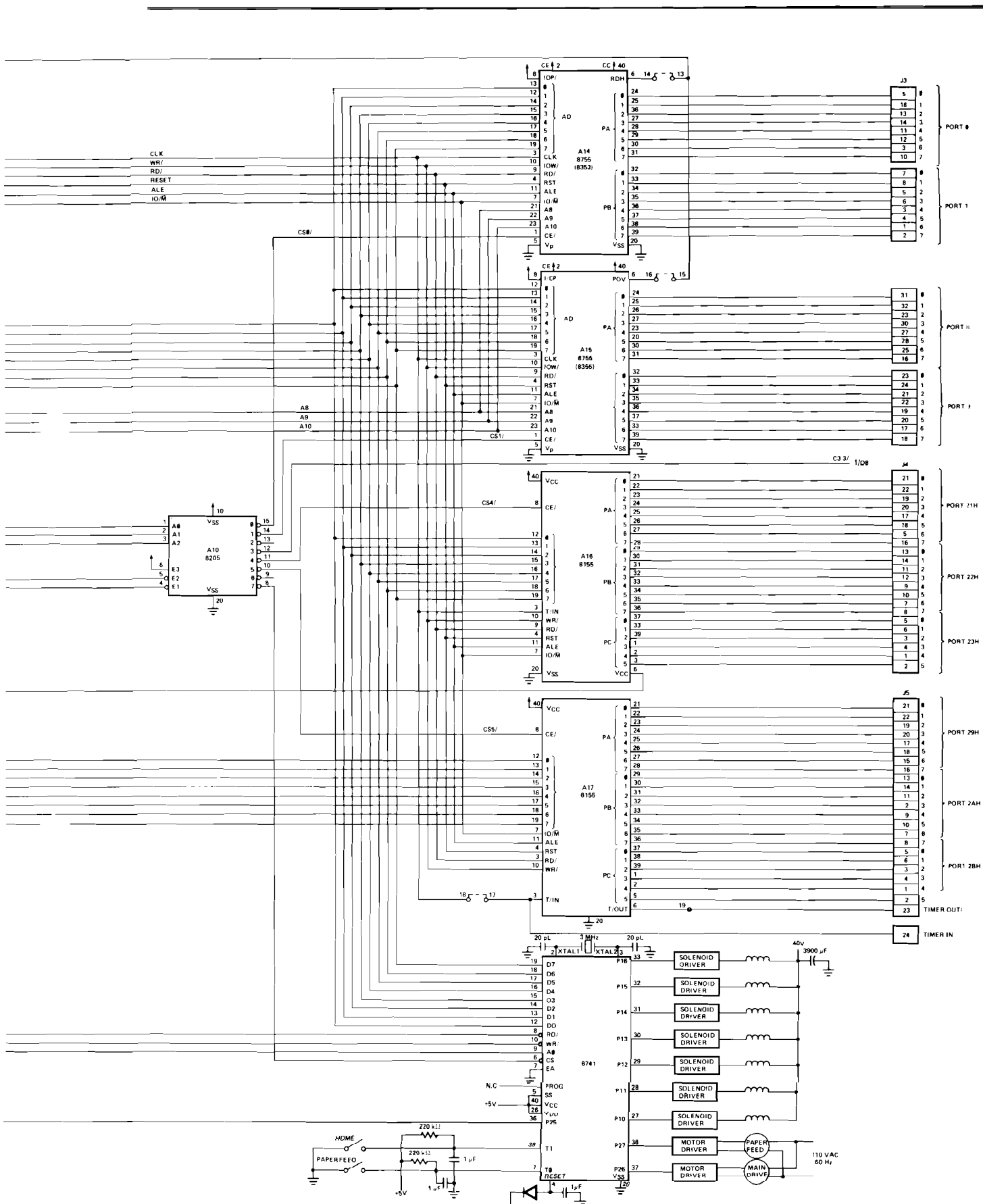


Figure 5. SDK-85 +UPI 41



TIMING

The relative timing of the interface signals to the printer is shown in Figure 6. Actual printing commences when the main drive switch signal goes into the print ready state. This edge indicates that the print head is scanning across the paper in the left to right direction and that the printer is ready to start the actual printing of characters. When this edge occurs the UPI-41 must start transmitting pulses to each of the seven solenoids. The timing for these pulses is shown on the last line of Figure 6. A pulse of about 400 microseconds is used to generate a dot on the paper; a pause of about 900 microseconds between these pulses satisfies the duty cycle restrictions of the solenoids and provides a space between dots. Since the printer does not provide any feedback to the UPI-41 which would indicate the position of the print head, it is necessary for the UPI-41 to decide when to fire each solenoid based on timing information it maintains internally. The specifications of the printer allow 310 milliseconds for the print head to traverse the print area. The maximum repetition rate at which the solenoids can be fired is once every 1.3 milliseconds. The maximum number of dots that can be printed in the available print area is then $310/1.3 = 238$. After the last dot has been printed the line feed motor can be activated. The motor should remain activated until the line feed switch makes the off to on to off transition; this takes about 200 milliseconds. After the line feed motor is deactivated the next time of interest is when the main drive signal goes to the inactive state. At this point the printing of a complete line, including the necessary line feed, has been accomplished and the ITPI-41 must prepare itself for the reactivation of the main drive switch. The activation of this switch will indicate that the printing of the next line can commence.

SOFTWARE

The software system necessary to drive the LRC printer can be thought of as two main parts, each with an associated data structure. A block diagram of the system is shown in Figure 7. All the items shown above the dotted line are associated with the BUFFER MANAGER (BMGR) program part. All items shown below the dotted line are associated with a PRINTER SERVICE ROUTINE (PSR).

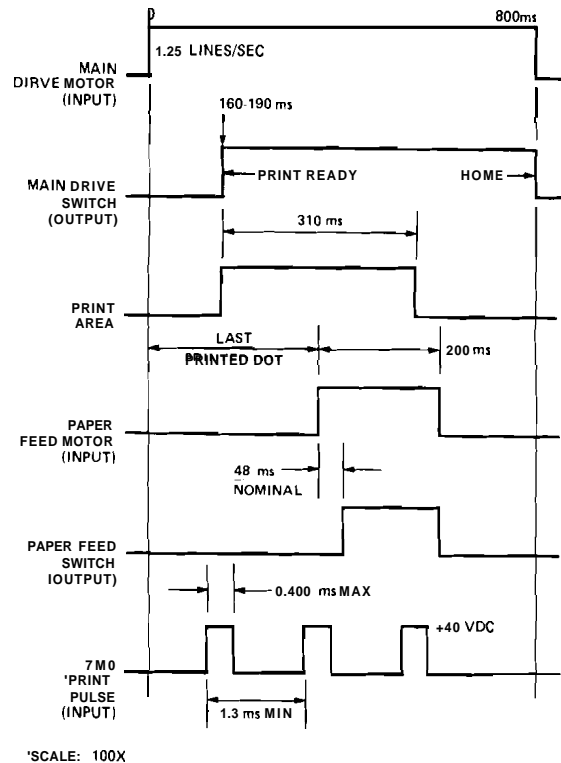


Figure 6. Printer Timing

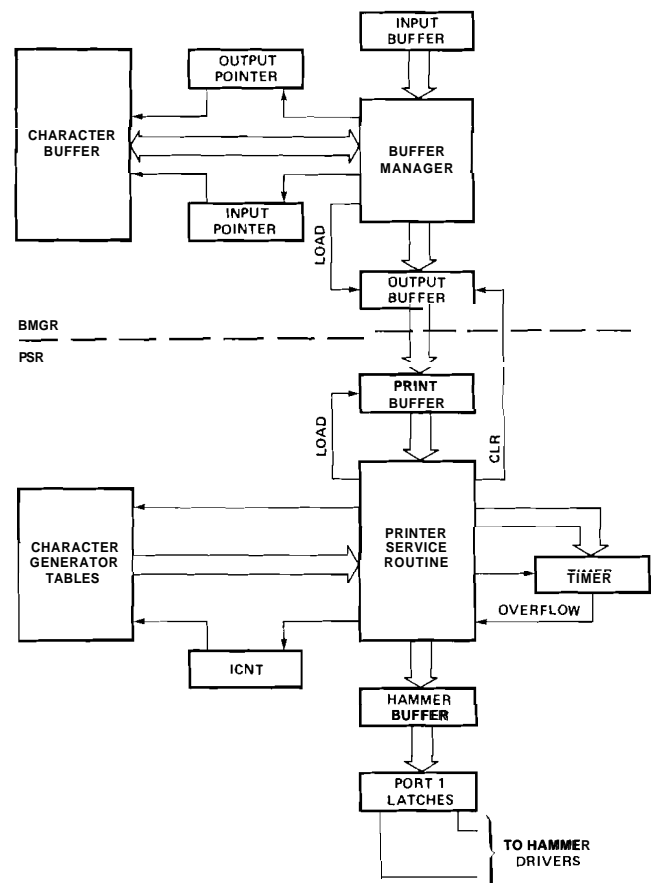


Figure 7. Software Block Diagram

The BUFFER MANAGER is responsible for all interaction with the master processor (i.e., the 8085 in Figure 5). The data structure associated with BMGR is a 40-character buffer which is used to store the characters as they are received from the master processor. BMGR maintains two pointers which are used to access the buffer; these pointers are shown as INPUT POINTER and OUTPUT POINTER in the diagram and are implemented as UPI-41 registers R₀ and R₁, respectively. The input pointer (INPNT) is kept pointing to the last character loaded into the buffer, the output pointer (OUTPNT) is kept pointing to the next character to be printed. BMGR has two major interfaces, the INPUT BUFFER, which is used to communicate with the master processor, and the register shown in the figure as OUTPUT BUFFER. This register, which is implemented with register R₃ of the UPI-41, is used to communicate with the printer service routine (PSR). A character to be printed is placed in the output buffer (OBUF). When PSR is ready to print the character it moves it from OBUF to its own buffer (PBUF) which is labeled as PRINT BUFFER in the diagram. After the character is moved the output buffer is overwritten by a predetermined value which indicates that PSR has accepted the character. BMGR will load a character into the output buffer only if it currently is equal to this value.

The printer service routine utilizes the TIMER to keep track of the current position of the print head. At the appropriate times it causes the solenoid drivers to be pulsed so that the character stream it sees in PBUF is printed. Based on the contents of PBUF and the contents of ICNT, which indicates the active column of the current character, PSR looks up the appropriate column data to be printed in the character generator tables. This data is stored in the HAMMER BUFFER until the precise time that it should be presented to the hammer drivers via the I/O bits in PORT 1. ICNT and the HAMMER BUFFER are implemented as UPI-41 registers 5 and 7, respectively.

DETAILS OF THE BUFFER MANAGER

Before BMGR can be discussed in detail, the manner in which it utilizes the character buffer must be understood. Figure 8 shows the operation of the buffer while two lines of data are input to the UPI-41 and subsequently printed. In order to keep the discussion manageable, this figure is drawn as if the printer were capable of printing only four

characters per line. The two lines of characters to be printed are:

ABCD
1234

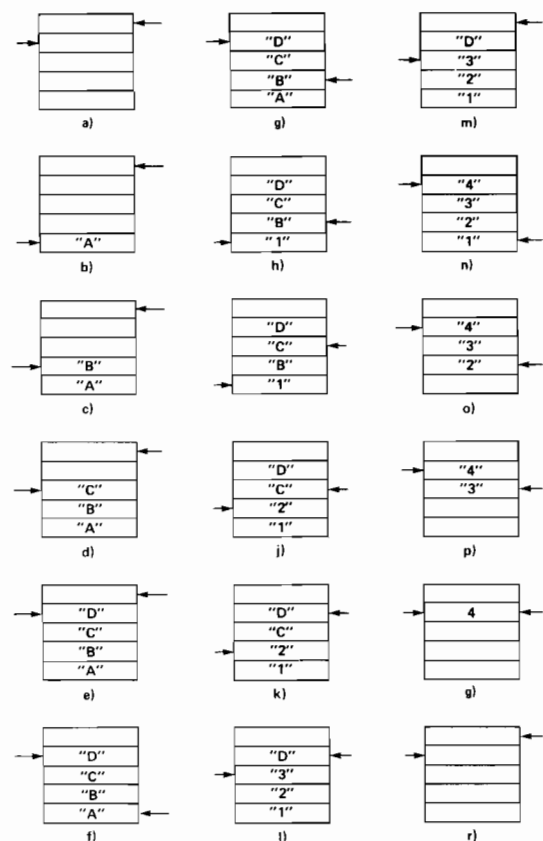


Figure 8. Buffer Operation

It should be noted that the buffer contains 5 bytes, one more than the number of print positions. The extra byte is a "phantom address" which, when pointed to by the output pointer, indicates that the section of BMGR which services the printer service routine is inactive. This state must be allowed because the actual print operation cannot begin until the complete line has been input to the buffer. If this rule were not enforced, some under-run protocol would have to be established to handle the situation of the input stream from the master processor failing to keep up with the print head.

Figure 8a shows the buffer in its initial state. The input pointer is set to the last real position in the buffer and the output pointer is set to the phantom position. Figures 8b through 8f show the operation of the pointers as the characters "A", "B", "C", and "D" are loaded. In each case the

input pointer is incremented to point to the next available location and then that location is loaded with the character. The position of the output pointer is not changed until the last position of the buffer has been loaded. When this occurs, the output pointer is set to point at the first character of the buffer. The operation of the pointers thus far can be described by the following algorithm:

```
INITIAL:
  INPOINT:=BUFFER_MAX;
  OUTPOINT:=BUFFER_MAX+1;
  ...
LOOP:
  IF CHARACTER_AVAILABLE THEN
  BEGIN
    INPOINT:=(INPOINT+1) MOD BUFFER_LENGTH;
    BUFFER(INPOINT):=CHARACTER;
    IF INPOINT=BUFFER_MAX THEN OUTPOINT:=BUFFER_MIN;
  END;
  GOTO LOOP;
END;
```

Obviously, if this loop were allowed to continue, the buffer would be overwritten by the next line of text before the first could be printed. This can be prevented by modifying the algorithm as follows:

```
...
LOOP:
  IF CHARACTER_AVAILABLE THEN
  BEGIN
    TEMP:=(INPOINT+1) MOD BUFFER_LENGTH;
    IF TEMP<>OUTPOINT THEN
    BEGIN
      INPOINT:=TEMP;
      BUFFER(INPOINT):=CHARACTER;
      IF INPOINT=BUFFER_MAX THEN OUTPOINT:=BUFFER_MIN;
    END;
  END;
  GOTO LOOP;
```

This modification will “freeze the action” at Figure 8f until the output pointer is incremented. When this occurs the input procedure will immediately load the input data over the character that was just printed (assuming that data is available to the procedure at a higher rate than can be printed). The defined interface with the printer service routine allows a character to be removed from the buffer and placed in the output buffer whenever the output buffer contains the value placed there by the PSR, indicating that it has accepted the character that was previously in the output buffer. If this value is called EMPTY_FLAG then the complete buffer handling procedure can be defined as follows:

```
INITIAL:
  INPOINT:=BUFFER_MAX;
  OUTPOINT:=BUFFER_MAX+1;
  ...
LOOP:
  IF CHARACTER_AVAILABLE THEN
  BEGIN
    TEMP:=(INPOINT+1) MOD BUFFER_LENGTH;
    IF TEMP<>OUTPOINT THEN
    BEGIN
      INPOINT:=TEMP;
      BUFFER(INPOINT):=CHARACTER;
      IF INPOINT=BUFFER_MAX THEN
        OUTPOINT:=BUFFER_MIN;
    END;
    IF OUTPUT_BUFFER=EMPTY_FLAG THEN
    BEGIN
      IF OUTPOINT<=BUFFER_MAX THEN
      BEGIN
        OUTPUT_BUFFER:=BUFFER(OUTPOINT);
        OUTPOINT:=OUTPOINT+1;
      END;
    END;
  END;
  GOTO LOOP;
```

Examination of Figures 8g through 8r will show how this algorithm maintains the buffer. If there is an open position and a character is available, it is placed in the buffer. When a complete line is in the buffer, printing is initialized by setting the output pointer to BUFFER_MIN. As the last character of a line is printed, the output pointer is incremented to point at the “phantom location” until the next line is completely entered. It should also be noted that if the input stream is faster than the print operation, then after the last character of a line is printed only one character need be input before printing can resume (see Figures 8l, m, and n). Frame r shows that after all available characters have been printed the state of the buffer is the same as it is initially. This is obviously a desirable feature.

The flowcharts for the complete BUFFER MANAGER are shown in Figures 9a and 9b. The corresponding code can be found starting at label BMGR of the program listings (see appendix). The flowcharts follow the algorithm that has been discussed very closely. Some additions have been made to implement logic not associated with the buffer. The first difference is that when a byte is in the input buffer it is tested to determine whether it is a command byte or a data character before further action is taken. Only two commands are recognized; one to set, and one to reset, the internal interrupt enable flag. This flag, which is

implemented as bit zero of PORT2 determines whether or not the UPI-41 will assert an interrupt to the master processor when it is able to accept a new character. Two additional deviations can be noted in Figure 9a; the first is that the motor of the printer will be turned on whenever a data character is received, the second is that if an end of line code (i.e., an ASCII line feed) is received, then, instead of storing it in the buffer, a mode is entered which fills the remaining buffer locations with space characters. This mode is enabled by bit one of PORT2. Note that utilizing otherwise unused bits of PORT2 for program status allows convenient testing and setting by the software and also enables external monitoring of the program operation.

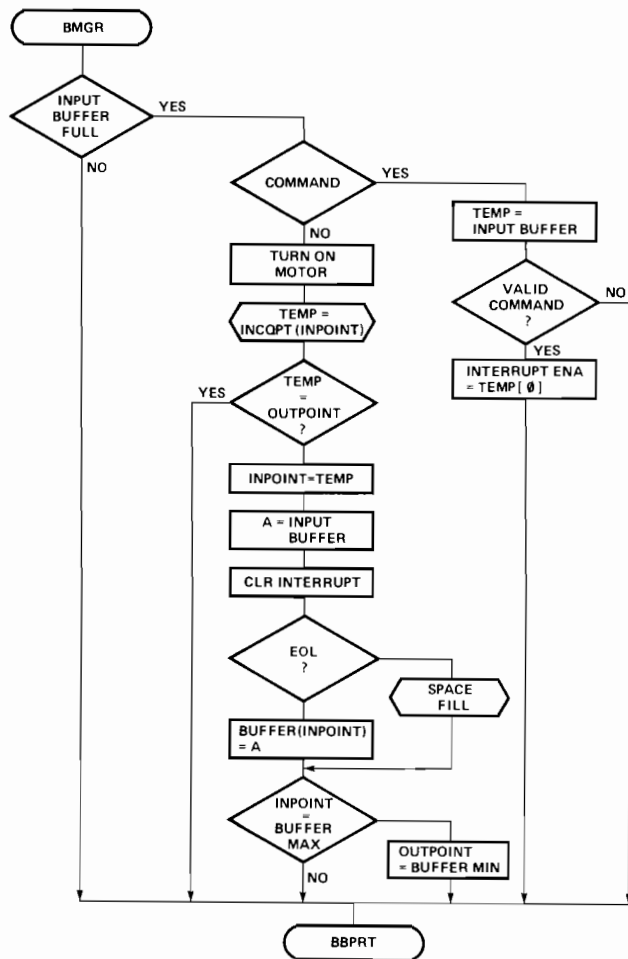


Figure 9a. Buffer Manager Flowchart

The last addition to the algorithm can be seen in Figure 9b where instead of going directly back to the start of the program after servicing the printer, a test is made to determine if the interrupt to the master processor should be asserted. This interrupt is set if the enable bit is set and there is also room in the buffer for at least one more character. After this test, control is passed back to the beginning of BMGR.

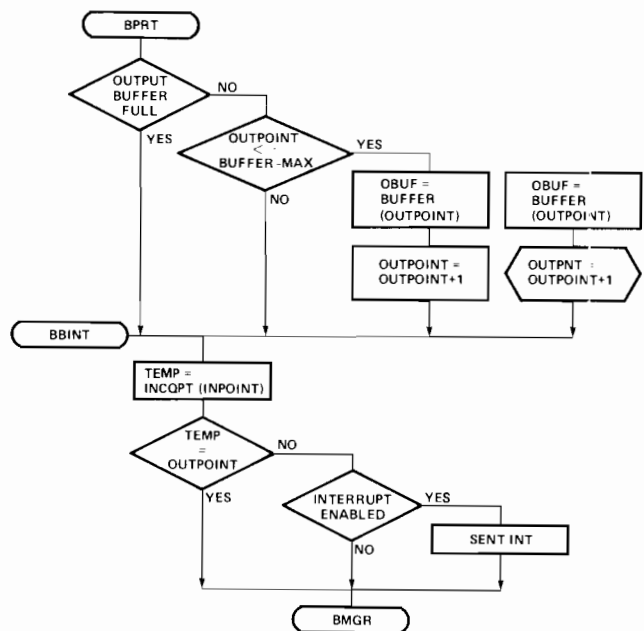


Figure 9b. Buffer Manager Flowchart

PRINTER SERVICE ROUTINES

The Printer Service Routine must convert the characters given to it by the Buffer Manager into an appropriately timed stream of pulses to the solenoids. Because the PSR is extremely time-dependent, it was implemented as an interrupt-driven routine which is given control when the timer overflow occurs. This allows exact timing of the solenoid firings without requiring software delay loops. If the timing had been generated by such loops, synchronization would have been lost when the delay loops were interrupted in order to service the master processor.

If a hardware design of a controller for the printer were being undertaken, a convenient place to start would be to generate a state transition diagram which shows all the states that can be entered and how control can transfer from state to state. This hardware design technique is often useful in software design and was, in fact, used to develop the PSR. The state diagram of the PSR is shown in Figure 10. A total of eight states are necessary to implement the printer control function. Before discussing this diagram further, each of these states must be defined.

- WPA:** The WPA (Wait for Print Area) state is the state in which the system waits for the input from the printer which indicates that it is ready to start the actual printing of data.
- TPA:** During the TPA (Test Print Area) state the system digitally filters the signal from the printer to ensure that contact bounce is not causing an erroneous indication that the print area has started.
- IPO:** Transfer to the IPO (Initialize Print Operation) state occurs after the positioning of the print head over the print area has been verified. During this state the system initializes itself to start printing a line of text.
- ICOL:** The ICOL (Inter Column) state is used to time the period between the activation of the hammers. During this state the space between the dots of the characters is generated.

PCOL: During the PCOL (Print Column) state the hammers are energized if the particular character being printed requires a dot in the corresponding position.

ICHAR: The ICHAR (Inter Character) state is active between characters on a given line.

WFON: During the WFON (Wait for Feed On) state the system waits for the assertion of the feed pulse from the printer. This signal indicates that the process of feeding paper is occurring.

WFOFF: The system remains in the WFOFF (Wait for Feed Off) until the feed pulse goes inactive. This indicates that the required paper feed operation has been completed.

The state diagram, in addition to defining the allowable states, also defines how state to state transitions can be made. The general structure of this diagram shows that PSR is initiated by the occurrence of the timer overflow interrupt. When the interrupt occurs the contents of the HAMDAT (HAMmer DATa) register are immediately transferred to PORT1 which causes the hammer solenoids to be activated. Each of the eight possible states sets data into the register which should be output at the next timer overflow occurrence and starts the timer operating in a mode which will result in the main program (BMGR) being interrupted at the proper time. The following paragraphs describe the operation of each of the states

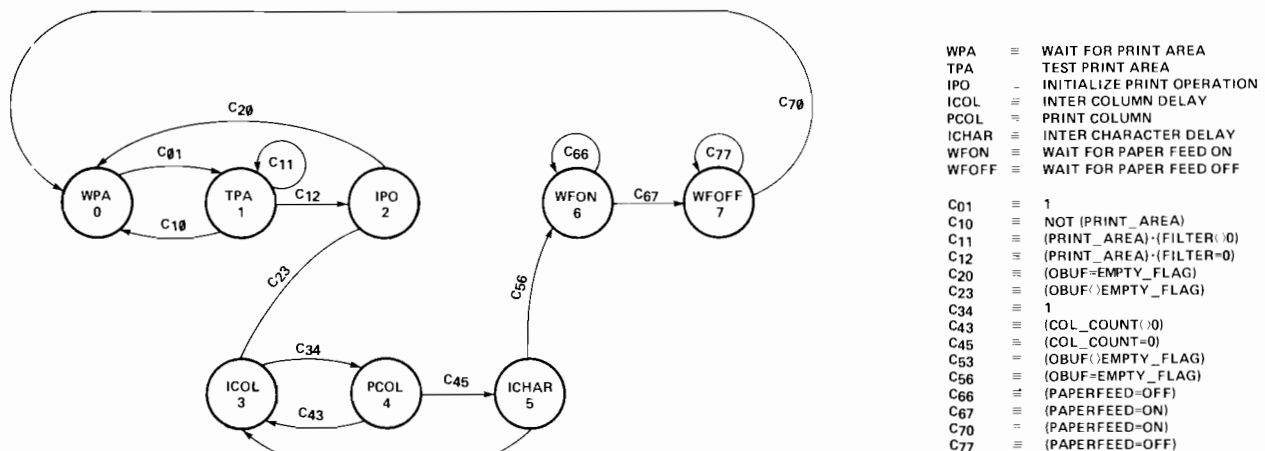


Figure 10. Print Control State Transition Diagram

in detail. The flowcharts of the routines can be found in Figure 11.

The WPA, CPA, and IPO states are all associated with the detection of the valid start of the print area. The WPA state sets the timer in the event count mode so that the edge of the print area signal can be detected, the CPA state digitally filters this input once it has been detected to ensure that noise has not caused a false input, and finally, the IPO state initializes the system to start the actual printing of data. The flowchart shows that the WPA state accomplishes the following actions:

1. Turns off the paper feed motor
2. Sets the filter count (for the CPA state)
3. Sets HAMDAT to zero
4. Sets STATE to one.

The timer is set to event count with an initial value of OFFH. This will cause a timer overflow interrupt the next time a negative transition occurs on the TEST1 input. Since this input is tied to the signal from the PRINT AREA switch, this interrupt should occur when the start of the print area is reached. The WPA state sets the STATE register to cause the TPA state to be entered when this interrupt occurs. Each time the TPA (Test Print Area) state is activated the software checks to ensure that the print area switch is in the proper state; if it is not, then all the actions of state zero are repeated (except turning off the motor), since a false start of print area has occurred. If the test reveals that the print area switch is in the proper state, then the filter count is reduced by one and the timer is started with an initial value of OFFH, the minimum attainable timer increment. The STATE register is set to repeat the TPA state unless the filter count has reached zero; when this occurs the IPO state is selected. The IPO state, which is responsible for the initialization of the actual print operation, first tests the output buffer register to determine if there is any data for it to print. If this test is unsuccessful the printer main drive motor is turned off, the TPA state is reinvoked and the timer is started in the event count mode so that it can detect the next start of print area. At first glance this seems somewhat fruitless since the event required cannot happen if the motor is not turning. By referring back to Figure 9, however, it can be seen that BMGR turns on the motor whenever it has a data character from the master computer. The reception of a character will always allow the PSR to find the next print area. If, when the IPO state makes its

test, there is data in the output buffer then the data is moved to the print buffer and the output buffer is set to the empty value. After this is accomplished, a counter is set to the number of columns to be printed per character (seven in this case — see comment by CGEN label in program listing), the STATE register is set to the ICOL state and the timer is set to time the intercolumn time. (The intercolumn time is the time that elapses between each possible column of the character.) Before exiting from this state the first column of data for the hammers is generated by the COLUMN routine and placed in the HAMDAT register.

The three states already discussed set the printer up so that it is ready to print. The next three states are repeated sequentially until the entire line of data has been printed. The ICOL state is probably the simplest of the states. When it is invoked the hammers have just been fired by the entry into the PSR. All that the ICOL state does is to set the timer to time the proper duration of the hammer strikes, clear the HAMDAT register, and set the STATE register to the PCOL state. The PCOL state, only slightly more complicated than the ICOL state, first decrements the column count. If the end of a character is detected (count equal zero), the HAMDAT register is cleared and the STATE register is set to invoke the ICHAR state. If the end of a character is not detected then the COLUMN routine is again used to determine the next data to be sent to the hammers and the ICOL state is reinvoked. When the ICOL state is active two things can happen, depending on whether there is more data to print. If there is data in the output buffer then a series of actions similar to those of the IPO state occur to reinitialize the printing of a character; if there is no more data in the line then the paper feed motor is turned on, HAMDAT is cleared, and the STATE register is set to the WFON state. The timer is set for approximately one millisecond so that the state of the paper feed switch can be sampled periodically by the WFON and WFOFF states.

The WFON and WFOFF states continue to set the timer to the one millisecond sample rate, the WFON state reinvokes itself until the paper feed switch input is detected and then it invokes the WFOFF state. The WFOFF state reinvokes itself until the paper feed switch is detected in the off state and then invokes the WPA state. The sole purpose of the WFON and WFOFF states is to ensure that an off to on to off transition occurs on

the paper feed switch. When this criterion is satisfied the WPA state is invoked which first turns off the paper feed motor and then proceeds to print the next line of data.

CONCLUSION

The UPI-41 has been shown to be easily capable of controlling the LRC matrix printer with no external logic other than drivers and receivers. The program listings which implement the algorithms discussed are shown in Appendix A. It should be noted that no attempt has been made to minimize the amount of code in the program; the emphasis

was on clarity of operation and ease of implementation. A careful programmer should be able to significantly reduce the amount of code space needed, especially in the printer service routine which duplicates much code in each STATE. Even with this relatively loose coding the printer control function, including the complete character tables, easily fit within the memory available in the UPI-41. The extra room in memory could be used to implement such extra features as tabulation, printing prestored messages, or even limited graphic capabilities. The power and flexibility of the UPI-41 make such features easy to implement.

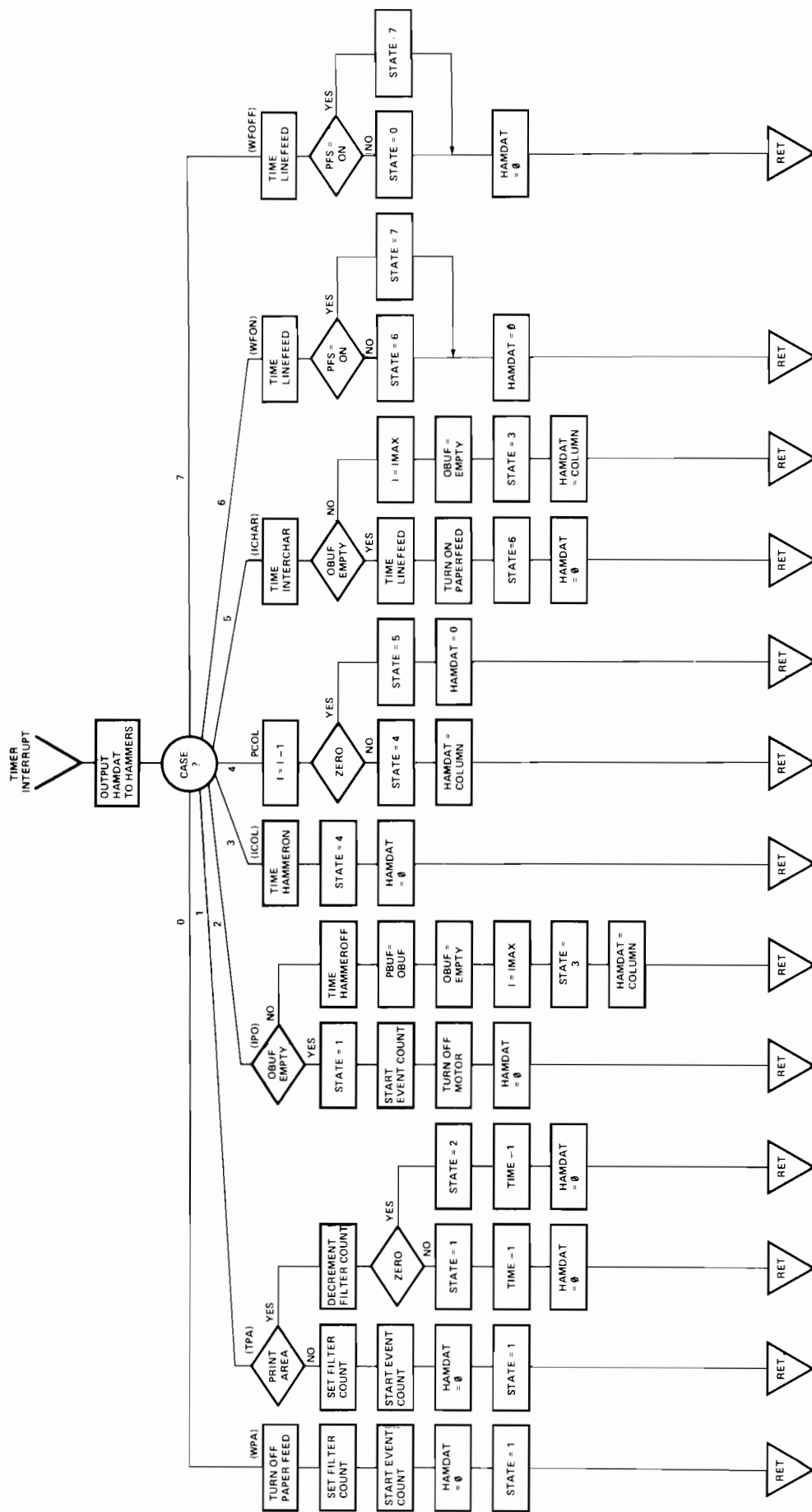


Figure 11. PSR Flowchart

APPENDIX

ISIS-II 8048 ASSEMBLER, V1.1

LRC PRINTER CONTROLLER 7/14/7

LOC	OBJ	SEQ	SOURCE STATEMENT
		1	
		2	
		3	*****
		4	;
		5	UPI-41 LRC PRINTER CONTROLLER
		6	;
		7	THIS PROGRAM IMPLEMENTS THE CONTROL OF THE
		8	LRC PRINTER WITH THE UPI-41. DATA IS INPUT TO THE
		9	UPI-41 AS SIX BIT ASCII. COMMANDS ARE PROVIDED
		10	TO ENABLE OR DISABLE THE GENERATION OF AN
		11	INTERRUPT WHEN THE UNIT IS READY
		12	FOR ANOTHER DATA CHARACTER. THE INTERRUPT IS ENABLED
		13	BY OUTPUTTING 03H TO THE CONTROL CHANNEL AND DISABLED
		14	BY OUTPUTTING 02H. WHEN ENABLED THE INTERRUPT
		15	IS IMPLEMENTED AS A POSITIVE GOING EDGE ON P25.
		16	;
		17	NOTE: A PL/M LIKE LANGUAGE WAS USED TO COMMENT
		18	THIS PROGRAM. NO COMPILER EXISTS FOR THE UPI-41.
		19	THE COMMENTS WERE 'HAND COMPILED' INTO UPI-41
		20	ASSEMBLY LANGUAGE.
		21	;
		22	*****
		23	
		24	
		25	
		26	*****
		27	;
		28	REGISTER ASSIGNMENTS
		29	;
		30	*****
		31	
		32	
0007		33	HAMDAT EQU R7
0006		34	STATE EQU R6
0005		35	ICNT EQU R5
0004		36	PBUF EQU R4
0003		37	OBUF EQU R3
0002		38	TESTR EQU R2
0001		39	OUTPNT EQU R1
0000		40	INPNT EQU R0
		41	
		42	
		43	
		44	*****
		45	;
		46	TIMER EQUATES
		47	;
		48	*****
		49	
00A0		50	TICK EQU 160
FFFE		51	THON EQU -320/TICK
FFFD		52	THOFF EQU -480/TICK
FFF8		53	TINTER EQU -1280/TICK
FFFA		54	TLFEED EQU -1000/TICK
0004		55	FILTV EQU 640/TICK
		56	
		57	
		58	*****
		59	;
		60	PROGRAM MASKS
		61	;
		62	*****
		63	
00FF		64	EMTFLG EQU 0FFH
0007		65	IMAX EQU 07H
007F		66	PFEED EQU 7FH
00BF		67	MOTON EQU 0BFH
0001		68	INTENA EQU 01H
0002		69	FMODE EQU 02H
000A		70	EOL EQU 0AH
0021		71	EXCLAIM EQU 021H
0020		72	SPACE EQU 20H
0020		73	EXREQ EQU 20H
0018		74	OPTMIN EQU 18H
0018		75	BMIN EQU 18H
003F		76	BMAX EQU 3FH
		77	
		78	
		79	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
		80	
		81	*****
		82	;
		83	;
		84	START OF PROGRAM
		85	*****
		86	
		87	;
		88	; INITIALIZE;
		89	; INITIALIZE AND GO TO
		90	; BMGR
0000		90	ORG 00H
0000	1416	91	RESET: CALL CASE0
0002	3479	92	CALL INIT
0004	25	93	EN TCNTI
0005	2400	94	JMP BMGR
		95	;
		96	; CODE MUST END AT LOC 6
		97	*****
		98	;
		99	;
		100	START OF INTERRUPT DRIVEN STATE MACHINE
		101	*****
		102	;
		103	;
		104	;
		105	DO;
		106	HAMMERS=HAMMERS\$DAT;
		107	DO CASE STATE;
0007	2F	106	TISR: XCH A,HAMDAT
0008	37	107	CPL A
0009	39	108	OUTL P1,A
000A	FE	109	MOV A,STATE
000B	030E	110	ADD A,#CBASE
000D	B3	111	JMPP @A
000E	16	112	CBASE: DB CASE0
000F	24	113	DB CASE1
0010	40	114	DB CASE2
0011	61	115	DB CASE3
0012	6B	116	DB CASE4
0013	7D	117	DB CASE5
0014	9E	118	DB CASE6
0015	AE	119	DB CASE7
		120	
		121	
		122	
		123	;
		124	;
		125	DO; /*CASE 0, FEEDING LINE */
		126	PAPERSFEED=OFF;
		127	STATE=1;
		128	ICNT=FILTV
		129	WAIT(PRINT\$AREA);
		130	HAMMERS\$DATA=0;
		131	END; /* END OF CASE 0 */
0016	8A80	132	CASE0: ORL P2,#(NOT PFEED)
0018	BE01	133	MOV STATE,#1
001A	BD04	134	MOV ICNT,#FILTV
001C	23FF	135	MOV A,#-1
001E	62	136	MOV T,A
001F	45	137	STRT CNT
0020	2300	138	MOV A,#0
0022	2F	139	XCH A,HAMDAT
0023	93	140	RETR
		141	;
		142	;
		143	DO; /* CASE1, TESTING FOR PRINT AREA */
		144	IF T0=1 THEN
		145	DO;
		146	STATE=1;
		147	ICNT=FILTV;
		148	WAIT(PRINT\$AREA);
		149	HAMMERS\$DATA=0;
		150	END;
0024	4632	150	CASE1: JNT1 C1ELS
0026	BE01	151	MOV STATE,#1
0028	BD04	152	MOV ICNT,#FILTV
002A	23FF	153	MOV A,#-1
002C	62	154	MOV T,A
002D	55	155	STRT T
002E	2300	156	MOV A,#0
0030	043E	157	JMP C1END

LOC	OBJ	SEQ	SOURCE STATEMENT	
		158		
		159		ELSE DO;
		160		ICNT=ICNT-1;
		161		IF ICNT=0 THEN STATE=2 ELSE STATE=1;
		162		TIME (-1);
		163		HAMMERSDATA=0;
0032	BE02	164	C1ELS: MOV STATE,#2	END;
0034	ED38	165	DJNZ ICNT,C1LA	
0036	BE01	166	MOV STATE,#1	
0038	23FF	167	C1LA: MOV A,#-1	
003A	62	168	MOV T,A	
003B	55	169	STRT T	
003C	2300	170	MOV A,#0	
003E	2F	171		END; /*END OF CASE1 */
003F	93	172	C1END: XCH A,HAMDAT	
		173	RETR	
		174		
		175		
		176		DO; /*CASE 2, INITIALIZE PRINT OPERATION */
		177		IF OBUF<>EMPTY\$FLAG THEN
		178		DO;
		179		TIME (HAMMERSOFF);
		180		PBUF=OBUF;
		181		OBUF=EMPTY\$FLAG;
		182		I=IMAX;
		183		STATE=3;
		184		HAMMERSDATA=COLUMN (PBUF,I);
		185		END;
0040	FB	186	CASE2: MOV A,OBUF	
0041	D3FF	187	XRL A,#EMTFLG	
0043	C655	188	JZ C2ELS	
0045	23FD	189	MOV A,#THOFF	
0047	62	190	MOV T,A	
0048	55	191	STRT T	
0049	FB	192	MOV A,OBUF	
004A	AC	193	MOV PBUF,A	
004B	BBFF	194	MOV OBUF,#EMTFLG	
004D	BD07	195	MOV ICNT,#IMAX	
004F	BE03	196	MOV STATE,#3	
0051	54E0	197	CALL COLUMN	
0053	045F	198	JMP C2END	
		199		ELSE DO;
		200		STATE=1;
		201		WAIT (PRINT\$AREA);
		202		MOTOR=OFF;
		203		HAMMERSDATA=0;
		204		END;
0055	BE01	205	C2ELS: MOV STATE,#1	
0057	23FF	206	MOV A,#-1	
0059	62	207	MOV T,A	
005A	45	208	STRT CNT	
005B	8A40	209	ORL P2,#NOT MOTON	
005D	2300	210	MOV A,#0	
005F	2F	211		END; /*END OF CASE 2 */
0060	93	212	C2END: XCH A,HAMDAT	
		213	RETR	
		214		
		215		DO; /*CASE 3, HAMMER OFF CYCLE */
		216		TIME (HAMMER\$ON);
		217		HAMMERSDATA=0;
		218		STATE=4;
		219		END; /*END OF CASE 3 */
0061	23FE	220	CASE3: MOV A,#THON	
0063	62	221	MOV T,A	
0064	55	222	STRT T	
0065	2300	223	MOV A,#0	
0067	BE04	224	MOV STATE,#4	
0069	2F	225	XCH A,HAMDAT	
006A	93	226	RETR	
		227	\$ EJECT	

LOC	OBJ	SEQ	SOURCE STATEMENT	
		228		DO; /*CASE 4, PRINTING COL I OF CHAR */
		229		TIME (HAMMER\$OFF);
006B	23FD	230	CASE4: MOV A, #THOFF	
006D	62	231	MOV T, A	
006E	55	232	STRT T	
		233		I=I-1;
		234		IF I=0 THEN
		235		DO;
		236		STATE=5;
		237		HAMMER\$DATA=0;
		238		END
006F	ED77	239	DJNZ ICNT, C4ELS	
0071	BE05	240	MOV STATE, #5	
0073	2300	241	MOV A, #0	
0075	047B	242	JMP C4END	
		243		ELSE DO;
		244		STATE=3;
		245		HAMMER\$DATA=COLUMN (PBUF, I);
		246		END;
0077	BE03	247	C4ELS: MOV STATE, #3	
0079	54E0	248	CALL COLUMN	
		249		END; /* END OF CASE 4 */
007B	2F	250	C4END: XCH A, HAMDAT	
007C	93	251	RETR	
		252		DO; /*CASE 5, INTERCHARACTER SPACE */
		253		TIME (INTER\$CHAR);
007D	23F8	254	CASE5: MOV A, #TINTER	
007F	62	255	MOV T, A	
0080	55	256	STRT T	
		257		IF OBUF<>EMPTY\$FLAG THEN
		258		DO;
		259		PBUF=OBUF;
		260		OBUF=EMPTY\$FLAG;
		261		I=IMAX;
		262		STATE=3;
		263		HAMMER\$DATA=COLUMN (PBUF, I);
		264		END;
0081	FB	265	MOV A, OBUF	
0082	D3FF	266	XRL A, #EMTFLG	
0084	C692	267	JZ C5ELS	
0086	FB	268	MOV A, OBUF	
0087	AC	269	MOV PBUF, A	
0088	BBFF	270	MOV OBUF, #EMTFLG	
008A	BD07	271	MOV ICNT, #IMAX	
008C	BE03	272	MOV STATE, #3	
008E	54E0	273	CALL COLUMN	
0090	049C	274	JMP C5END	
		275		ELSE DO;
		276		TIME (LINE\$FEED);
		277		PAPER\$FEED=ON;
		278		STATE=6;
		279		HAMMER\$DATA=0;
		280		END;
0092	23FA	281	C5ELS: MOV A, #TLFEED	
0094	62	282	MOV T, A	
0095	55	283	STRT T	
0096	9A7F	284	ANL P2, #PFEED	
0098	BE06	285	MOV STATE, #6	
009A	2300	286	MOV A, #0	
		287		END; /* END OF CASE 5 */
009C	2F	288	C5END: XCH A, HAMDAT	
009D	93	289	RETR	
		290		
		291	\$ EJECT	

LOC	OBJ	SEQ	SOURCE STATEMENT
		292	;
		293	;
		294	DO; /*CASE 6, WAITING FOR FEED ON */
		295	TIME (LINESFEED);
		296	IF PFS=1 THEN
		297	DO;
		298	STATE=7;
009E 23FA		299	END;
00A0 62		300	;
00A1 55		301	ELSE DO;
00A2 26A8		302	STATE=6;
00A4 BE07		303	END;
00A6 04AA		304	;
		305	;
		306	;
00A8 BE06		307	C6ELS: MOV STATE, #6
		308	;
		309	;
00AA 2300		310	C6END: MOV A, #0
00AC 2F		311	XCH A, HAMDAT
00AD 93		312	RETR
		313	;
		314	;
		315	DO; /*CASE 7, WAITING FOR FEED OFF */
		316	TIME (LINESFEED);
		317	IF PFS=0 THEN
		318	DO;
		319	STATE=0;
00AE 23FA		320	END;
00B0 62		321	;
00B1 55		322	ELSE DO;
00B2 36B8		323	STATE=7;
00B4 BE00		324	END;
00B6 04BA		325	;
		326	;
		327	;
		328	;
		329	;
		330	;
00B8 BE07		331	C7ELS: MOV STATE, #7
00BA 2300		332	C7END: MOV A, #0
00BC 2F		333	XCH A, HAMDAT
00BD 93		334	RETR
		335	;
		336	;
		337	END; /* END OF CASE BLOCK */
		338	;
		339	;
		340	;
		341	*****
		342	;
		343	BMGR
		344	;
		345	THIS SEGMENT CONTROLS THE HANDSHAKING BETWEEN THE
		346	CONTROLLER AND THE MASTER PROCESSOR.
		347	;
		348	*****
		349	;
		350	;
		351	;/BMGR-BUFFER MANAGER*/
		352	;
		353	DO;
		354	IF IBF=FULL THEN
		355	DO;
		356	IF TYPE=DATA THEN
		357	DO;
		358	MOTOR=ON;
		359	TEMP=INCQPT (INPNT);
		360	;
		361	;
0100		362	ORG 100H
		363	;
0100 D647		364	BMGR: JNIBF BBPRT
0102 7636		365	JF1 BBCMD
0104 9ABF		366	ANL P2, #MOTON
0106 F8		367	MOV A, INPNT
0107 3470		368	CALL INCQPT

LOC	OBJ	SEQ	SOURCE STATEMENT
		369	;
		370	;
		371	;
		372	IF TEMP<>OUT\$POINT THEN
		373	DO;
		374	IN\$POINT=TEMP;
		375	IF FILL\$MODE=ON THEN
		376	DO;
		377	TEMP=SPACE;
		378	ELSE DO;
		379	TEMP=INPUT\$BUFFER;
		380	INTERRUPT=OFF;
		381	END;
		382	IF TEMP=EOL THEN
		383	DO;
		384	FILL\$MODE=ON;
		385	TEMP=SPACE;
		386	END;
		387	IF TEMP=CONTROL\$CODE THEN TEMP='!';
		388	BUFFER(IN\$POINT)=TEMP AND 03FH;
		389	IF IN\$POINT=BUFFER\$MAX THEN
		390	DO;
		391	FILL\$MODE=OFF;
		392	OUT\$POINT=BUFFER\$MIN;
		393	END;
0109	D9	394	BBL1: XRL A,OUTPNT
010A	C647	395	JZ BBPRT
010C	D9	396	XRL A,OUTPNT
010D	A8	397	MOV INPNT,A
010E	0A	398	IN A,P2
010F	3216	399	JB1 FILL
0111	22	400	IN A,DBB
0112	9ADF	401	ANL P2,#NOT(EXREQ)
0114	2418	402	JMP BBL1A
0116	2320	403	FILL: MOV A,#SPACE
0118	D30A	404	BBL1A: XRL A,#EOL
011A	9620	405	JNZ BBL1B
011C	8A02	406	ORL P2,#FMODE
011E	2428	407	JMP BBL1C
0120	D30A	408	BBL1B: XRL A,#EOL
0122	D228	409	JB6 BBL1C
0124	B228	410	JB5 BBL1C
0126	2321	411	MOV A,#EXCLAIM
0128	533F	412	BBL1C: ANL A,#03FH
012A	A0	413	MOV @INPNT,A
012B	F8	414	MOV A,INPNT
012C	D33F	415	XRL A,#BMAX
012E	9647	416	JNZ BBPRT
0130	9AFD	417	ANL P2,#NOT FMODE
0132	B918	418	MOV OUTPNT,#BMIN
0134	2447	419	JMP BBPRT
		420	;
		421	ELSE DO; /*TYPE IS COMMAND*/
		422	INTERRUPT=OFF;
		423	IF (PORT0 AND 3)=2 THEN INTENA=OFF;
		424	IF (PORT0 AND 3)=3 THEN INTENA=ON;
		425	END;
0136	22	426	BBCMD: IN A,DBB
0137	9ADF	427	ANL P2,#NOT(EXREQ)
0139	5303	428	ANL A,#3
013B	323F	429	JB1 BBL2
013D	2447	430	JMP BBPRT
013F	1245	431	BBL2: JB0 BBL3
0141	9AFE	432	ANL P2,#NOT INTENA
0143	2447	433	JMP BBPRT
0145	8A01	434	BBL3: ORL P2,#INTENA
		435	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
		436	;
		437	IF OBUF=EMPTY\$FLAG AND (OUT\$POINT<>BMIN OR STATE
		438	DO;
		439	IF OUT\$POINT<=BUFFER\$MAX THEN
		440	DO;
		441	OBUF=BIN(OUT\$POINT);
		442	OUT\$POINT=OUT\$POINT+1;
		443	END;
		444	END;
0147	FB	445	BBPRT: MOV A,OBUF
0148	D3FF	446	XRL A,#EMTFLG
014A	965E	447	JNZ BINT
014C	F9	448	MOV A,OUTPNT
014D	D318	449	XRL A,#QPTMIN
014F	9658	450	JNZ BBPRTA
0151	FE	451	MOV A,STATE
0152	03FD	452	ADD A,#-3
0154	F258	453	JB7 BBPRTA
0156	245E	454	JMP BINT
0158	F9	455	BBPRTA: MOV A,OUTPNT
0159	D25E	456	JB6 BINT
015B	F1	457	MOV A,@OUTPNT
015C	AB	458	MOV OBUF,A
015D	19	459	INC OUTPNT
		460	;
		461	TEMP=INCQPT(INPNT);
		462	IF TEMP<>OUT\$POINT THEN
		463	DO;
		464	IF INTENA=ON AND FMODE=OFF THEN INTERRUPT=ON;
		465	END;
		466	END;
		467	END
015E	F8	468	BINT: MOV A,INPNT
015F	3470	469	CALL INCQPT
0161	D9	470	XRL A,OUTPNT
0162	C600	471	JZ BMGR
		472	BINTA:
0164	0A	473	IN A,P2
0165	37	474	CPL A
0166	1200	475	JB0 BMGR
0168	326C	476	JB1 SETINT
016A	2400	477	JMP BMGR
016C	8A20	478	SETINT: ORL P2,#EXREQ
016E	2400	479	JMP BMGR
		480	
		481	
		482	
		483	
		484	
		485	
		486	
		487	;
		488	PROCEDURE INCQPT(A,CARRY);
		489	DO;
		490	A=A MOD BUFFER LENGTH+BUFFER MIN;
		491	IF A=BUFFER_MIN THEN CARRY=1;
		492	END;
0170	0301	492	INCQPT: ADD A,#1
0172	D275	493	JB6 ADJUST
0174	83	494	RET
0175	2318	495	ADJUST: MOV A,#QPTMIN
0177	A7	496	CPL C
0178	83	497	RET
		498	
		499	
		500	;
		501	PROCEDURE INIT;
		502	DO;
		503	OBUF=EMPTY\$FLAG;
		504	OUT\$POINT=BUFFER\$MAX+1;
		505	IN\$POINT=BUFFER\$MAX;
		506	MOTOR=OFF;
		507	PAPER\$FEED=OFF;
		508	FILL\$MODE=OFF;
		509	INTERRUPT\$MASK=OFF;
		510	BUS\$BUFFER=0;
		511	END;
0179	BBFF	511	INIT: MOV OBUF,#EMTFLG
017B	B940	512	MOV OUTPNT,#BMAX+1
017D	B83F	513	MOV INPNT,#BMAX
017F	23F0	514	MOV A,#0F0H
0181	3A	515	OUTL P2,A
0182	22	516	IN A,DBB
0183	83	517	RET
		518	
		519	
		520	
		521	
		522	
		523	
		524	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
		525	*****
		526	;
		527	;
		528	;
		529	;
		530	;
		531	;
		532	;
		533	*****
		534	;
		535	;
02E0		536	ORG 2E0H
		537	;
		538	;
		539	;
		540	;
		541	;
		542	;
		543	;
		544	;
		545	;
		546	;
		547	;
		548	;
		549	;
		550	;
		551	;
		552	;
		553	;
		554	;
		555	;
		556	;
		557	;
		558	;
		559	;
		560	;
		561	;
		562	;
		563	;
		564	;
		565	;
		566	;
		567	;
		568	;
		569	;
		570	;
		571	;
		572	;
		573	;
		574	;
		575	;
		576	;
		577	*****
		578	;
		579	;
		580	;
		581	;
		582	;
		583	;
		584	;
		585	;
		586	;
		587	;
		588	;
		589	;
		590	;
		591	;
		592	;
		593	;
		594	;
		595	;
		596	;
		597	;
		598	;
		599	;
		600	*****

LOC	OBJ	SEQ	SOURCE	STATEMENT
		601		
0200		602	ORG	200H
		603		
		604		
0200	0C	605	DB	0CH ; ** ; [AT]
0201	22	606	DB	22H ; * *
0202	41	607	DB	41H ; * *
0203	58	608	DB	58H ; * *
0204	01	609	DB	01H ; *
0205	48	610	DB	48H ; *
0206	00	611	DB	00H ;
		612		
0207	0F	613	DB	0FH ; **** ; [A]
0208	10	614	DB	10H ; *
0209	24	615	DB	24H ; *
020A	40	616	DB	40H ; *
020B	24	617	DB	24H ; *
020C	10	618	DB	10H ; *
020D	0F	619	DB	0FH ; ****
		620		
		621		
020E	7F	622	DB	7FH ; **** ; [B]
020F	00	623	DB	00H ;
0210	49	624	DB	49H ; *
0211	00	625	DB	00H ; *
0212	08	626	DB	08H ; *
0213	55	627	DB	55H ; *
0214	22	628	DB	22H ; *
		629		
0215	3E	630	DB	3EH ; **** ; [C]
0216	41	631	DB	41H ; *
0217	00	632	DB	00H ; *
0218	41	633	DB	41H ; *
0219	00	634	DB	00H ; *
021A	41	635	DB	41H ; *
021B	22	636	DB	22H ; *
		637		
021C	7F	638	DB	7FH ; **** ; [D]
021D	00	639	DB	00H ;
021E	41	640	DB	41H ; *
021F	00	641	DB	00H ; *
0220	00	642	DB	00H ; *
0221	41	643	DB	41H ; *
0222	3E	644	DB	3EH ; ****
		645		
0223	7F	646	DB	7FH ; **** ; [E]
0224	00	647	DB	00H ;
0225	49	648	DB	49H ; *
0226	00	649	DB	00H ; *
0227	49	650	DB	49H ; *
0228	00	651	DB	00H ; *
0229	41	652	DB	41H ; *
		653		
022A	7F	654	DB	7FH ; **** ; [F]
022B	00	655	DB	00H ;
022C	48	656	DB	48H ; *
022D	00	657	DB	00H ; *
022E	48	658	DB	48H ; *
022F	00	659	DB	00H ; *
0230	40	660	DB	40H ; *
		661		
0231	3E	662	DB	3EH ; **** ; [G]
0232	41	663	DB	41H ; *
0233	00	664	DB	00H ; *
0234	41	665	DB	41H ; *
0235	04	666	DB	04H ; *
0236	41	667	DB	41H ; *
0237	26	668	DB	26H ; ** *
		669		
		670	\$	EJECT

LOC	OBJ	SEQ	SOURCE	STATEMENT
0238	7F	671	DB	7FH ; ***** ; [H]
0239	00	672	DB	00H ;
023A	08	673	DB	08H ; *
023B	00	674	DB	00H ;
023C	08	675	DB	08H ; *
023D	00	676	DB	00H ;
023E	7F	677	DB	7FH ; *****
		678		
023F	00	679	DB	00H ;
0240	41	680	DB	41H ; * * ; [I]
0241	00	681	DB	00H ;
0242	7F	682	DB	7FH ; *****
0243	00	683	DB	00H ;
0244	41	684	DB	41H ; * *
0245	00	685	DB	00H ;
		686		
0246	02	687	DB	02H ; * ; [J]
0247	01	688	DB	01H ; *
0248	00	689	DB	00H ;
0249	01	690	DB	01H ; *
024A	00	691	DB	00H ;
024B	01	692	DB	01H ; *
024C	7E	693	DB	7EH ; *****
		694		
024D	7F	695	DB	7FH ; ***** ; [K]
024E	00	696	DB	00H ;
024F	04	697	DB	04H ; *
0250	14	698	DB	14H ; * *
0251	22	699	DB	22H ; * *
0252	41	700	DB	41H ; * *
0253	00	701	DB	00H ;
		702		
0254	7F	703	DB	7FH ; ***** ; [L]
0255	00	704	DB	00H ;
0256	01	705	DB	01H ; *
0257	00	706	DB	00H ;
0258	01	707	DB	01H ; *
0259	00	708	DB	00H ;
025A	01	709	DB	01H ; *
		710		
025B	7F	711	DB	7FH ; ***** ; [M]
025C	40	712	DB	40H ; *
025D	20	713	DB	20H ; *
025E	18	714	DB	18H ; **
025F	20	715	DB	20H ; *
0260	40	716	DB	40H ; *
0261	3F	717	DB	3FH ; *****
		718		
		719		
0262	7F	720	DB	7FH ; ***** ; [N]
0263	20	721	DB	20H ; *
0264	10	722	DB	10H ; *
0265	08	723	DB	08H ; *
0266	04	724	DB	04H ; *
0267	00	725	DB	00H ;
0268	7F	726	DB	7FH ; *****
		727		
0269	3E	728	DB	3EH ; ***** ; [O]
026A	41	729	DB	41H ; * *
026B	00	730	DB	00H ;
026C	41	731	DB	41H ; * *
026D	00	732	DB	00H ;
026E	41	733	DB	41H ; * *
026F	3E	734	DB	3EH ; *****
		735		
0270	37	736	DB	37H ; *** ** ; [P]
0271	00	737	DB	00H ;
0272	48	738	DB	48H ; * *
0273	00	739	DB	00H ;
0274	00	740	DB	00H ;
0275	48	741	DB	48H ; * *
0276	30	742	DB	30H ; **
		743		
0277	3E	744	DB	3EH ; ***** ; [Q]
0278	41	745	DB	41H ; * *
0279	00	746	DB	00H ;
027A	40	747	DB	40H ; *
027B	05	748	DB	05H ; * *
027C	42	749	DB	42H ; * *
027D	3D	750	DB	3DH ; * *****

LOC	OBJ	SEQ	SOURCE STATEMENT
027E	7F	751	
027F	00	752	DB 7FH ; ***** ; [R]
0280	48	753	DB 00H ; ;
0281	00	754	DB 48H ; ; * *
0282	04	755	DB 00H ; ; *
0283	4A	756	DB 04H ; ; * * *
0284	31	757	DB 4AH ; ; * **
		758	DB 31H ; ; *
		759	
0285	32	760	DB 32H ; ; * ** ; [S]
0286	49	761	DB 49H ; ; * * *
0287	00	762	DB 00H ; ; *
0288	49	763	DB 49H ; ; * * *
0289	00	764	DB 00H ; ; *
028A	49	765	DB 49H ; ; * * *
028B	26	766	DB 26H ; ; ** *
		767	
		768	
028C	40	769	DB 40H ; ; * ; [T]
028D	00	770	DB 00H ; ; *
028E	40	771	DB 40H ; ; *
028F	3F	772	DB 3FH ; ; *****
0290	40	773	DB 40H ; ; *
0291	00	774	DB 00H ; ; *
0292	40	775	DB 40H ; ; *
		776	
0293	7C	777	DB 7CH ; ; ***** ; [U]
0294	02	778	DB 02H ; ; *
0295	01	779	DB 01H ; ; *
0296	00	780	DB 00H ; ; *
0297	01	781	DB 01H ; ; *
0298	02	782	DB 02H ; ; *
0299	7C	783	DB 7CH ; ; *****
		784	
029A	78	785	DB 78H ; ; ***** ; [V]
029B	04	786	DB 04H ; ; *
029C	02	787	DB 02H ; ; *
029D	01	788	DB 01H ; ; *
029E	02	789	DB 02H ; ; *
029F	04	790	DB 04H ; ; *
02A0	78	791	DB 78H ; ; *****
		792	
02A1	7E	793	DB 7EH ; ; ***** ; [W]
02A2	01	794	DB 01H ; ; *
02A3	02	795	DB 02H ; ; *
02A4	0C	796	DB 0CH ; ; **
02A5	02	797	DB 02H ; ; *
02A6	01	798	DB 01H ; ; *
02A7	7E	799	DB 7EH ; ; *****
		800	
02A8	41	801	DB 41H ; ; * * * ; [X]
02A9	22	802	DB 22H ; ; * * *
02AA	14	803	DB 14H ; ; * * *
02AB	08	804	DB 08H ; ; * * *
02AC	14	805	DB 14H ; ; * * *
02AD	22	806	DB 22H ; ; * * *
02AE	41	807	DB 41H ; ; * * *
		808	
02AF	40	809	DB 40H ; ; * ; [Y]
02B0	20	810	DB 20H ; ; *
02B1	10	811	DB 10H ; ; *
02B2	0F	812	DB 0FH ; ; *****
02B3	10	813	DB 10H ; ; *
02B4	20	814	DB 20H ; ; *
02B5	40	815	DB 40H ; ; *
		816	
		817	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
02B6	41	818	DB 41H ; * * ; [Z]
02B7	02	819	DB 02H ; * ;
02B8	45	820	DB 45H ; * * * ;
02B9	08	821	DB 08H ; * * ;
02BA	51	822	DB 51H ; * * * ;
02BB	20	823	DB 20H ; * * ;
02BC	41	824	DB 41H ; * * ;
02BD	7F	825	
02BE	00	826	DB 7FH ; ***** ; [I]
02BF	41	827	DB 00H ; * * ;
02C0	00	828	DB 41H ; * * ;
02C1	41	829	DB 00H ; * * ;
02C2	00	830	DB 41H ; * * ;
02C3	41	831	DB 00H ; * * ;
02C4	40	832	DB 41H ; * * ;
02C5	20	833	
02C6	10	834	DB 40H ; * * ; [N]
02C7	08	835	DB 20H ; * * ;
02C8	04	836	DB 10H ; * * ;
02C9	02	837	DB 08H ; * * ;
02CA	01	838	DB 04H ; * * ;
02CB	41	839	DB 02H ; * * ;
02CC	00	840	DB 01H ; * * ;
02CD	41	841	
02CE	00	842	DB 41H ; * * ; [I]
02CF	41	843	DB 00H ; * * ;
02D0	00	844	DB 41H ; * * ;
02D1	7F	845	DB 00H ; * * ;
02D2	00	846	DB 41H ; * * ;
02D3	04	847	DB 00H ; ***** ;
02D4	08	848	
02D5	10	849	DB 00H ; * * ; [UA]
02D6	08	850	DB 04H ; * * ;
02D7	04	851	DB 08H ; * * ;
02D8	00	852	DB 10H ; * * ;
02D9	01	853	DB 08H ; * * ;
02DA	00	854	DB 04H ; * * ;
02DB	01	855	DB 00H ; * * ;
02DC	00	856	
02DD	01	857	DB 01H ; * * ; []
02DE	00	858	DB 00H ; * * ;
02DF	01	859	DB 01H ; * * ;
		860	DB 00H ; * * ;
		861	DB 01H ; * * ;
		862	DB 00H ; * * ;
		863	DB 01H ; * * ;
		864	DB 00H ; * * ;
		865	
		866	
		867	
		868	
		869	*****
		870	*****
		871	START OF SECOND HALF OF CGEN TABLE
		872	*****
		873	*****
		874	
0300		875	ORG 300H
		876	
0300	00	877	
0301	00	878	DB 00H ; ; []
0302	00	879	DB 00H ; ;
0303	00	880	DB 00H ; ;
0304	00	881	DB 00H ; ;
0305	00	882	DB 00H ; ;
0306	00	883	DB 00H ; ;
		884	DB 00H ; ;
		885	
0307	00	886	DB 00H ; ; [!]
0308	00	887	DB 00H ; ;
0309	00	888	DB 00H ; ;
030A	7D	889	DB 7DH ; * *****
030B	00	890	DB 00H ; ;
030C	00	891	DB 00H ; ;
030D	00	892	DB 00H ; ;
		893	
		894	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
030E 00		895	DB 00H ; ; ["
030F 20		896	DB 20H ; ; *
0310 40		897	DB 40H ; ; *
0311 00		898	DB 00H ; ; *
0312 20		899	DB 20H ; ; *
0313 40		900	DB 40H ; ; *
0314 00		901	DB 00H ; ;
		902	
0315 14		903	DB 14H ; * * ; [#]
0316 00		904	DB 00H ; ;
0317 7F		905	DB 7FH ; *****
0318 00		906	DB 00H ; ;
0319 7F		907	DB 7FH ; *****
031A 00		908	DB 00H ; ;
031B 14		909	DB 14H ; * *
		910	
031C 00		911	DB 00H ; ; [\$]
031D 32		912	DB 32H ; * **
031E 49		913	DB 49H ; * * *
031F 36		914	DB 36H ; * * *
0320 49		915	DB 49H ; * * *
0321 26		916	DB 26H ; * * *
0322 00		917	DB 00H ; ;
		918	
0323 51		919	DB 51H ; * * * ; [%]
0324 02		920	DB 02H ; * * *
0325 54		921	DB 54H ; * * *
0326 08		922	DB 08H ; * * *
0327 15		923	DB 15H ; * * *
0328 20		924	DB 20H ; * * *
0329 45		925	DB 45H ; * * *
		926	
032A 26		927	DB 26H ; * * * ; [&]
032B 49		928	DB 49H ; * * *
032C 10		929	DB 10H ; * * *
032D 49		930	DB 49H ; * * *
032E 26		931	DB 26H ; * * *
032F 01		932	DB 01H ; * * *
0330 05		933	DB 05H ; * *
		934	
0331 00		935	DB 00H ; ; [']
0332 00		936	DB 00H ; ;
0333 10		937	DB 10H ; ; *
0334 20		938	DB 20H ; ; *
0335 40		939	DB 40H ; ; *
0336 00		940	DB 00H ; ;
0337 00		941	DB 00H ; ;
		942	
		943	
0338 1C		944	DB 1CH ; * * * ; [(
0339 22		945	DB 22H ; * * *
033A 41		946	DB 41H ; * * *
033B 00		947	DB 00H ; ;
033C 00		948	DB 00H ; ;
033D 00		949	DB 00H ; ;
033E 00		950	DB 00H ; ;
		951	
033F 00		952	DB 00H ; ; [)]
0340 00		953	DB 00H ; ;
0341 00		954	DB 00H ; ;
0342 00		955	DB 00H ; ;
0343 41		956	DB 41H ; * * *
0344 22		957	DB 22H ; * * *
0345 1C		958	DB 1CH ; * * *
		959	
0346 49		960	DB 49H ; * * * ; [*]
0347 22		961	DB 22H ; * * *
0348 1C		962	DB 1CH ; * * *
0349 77		963	DB 77H ; * * *
034A 1C		964	DB 1CH ; * * *
034B 22		965	DB 22H ; * * *
034C 49		966	DB 49H ; * * *
		967	
034D 08		968	DB 08H ; * ; [+]
034E 08		969	DB 08H ; * ;
034F 08		970	DB 08H ; * ;
0350 3E		971	DB 3EH ; * * * *
0351 08		972	DB 08H ; * ;
0352 08		973	DB 08H ; * ;
0353 08		974	DB 08H ; * ;

LOC	OBJ	SEQ	SOURCE STATEMENT
		975	
0354	00	976	DB 00H ; ; [,]
0355	00	977	DB 00H ; ;
0356	00	978	DB 00H ; ;
0357	01	979	DB 01H ; * ;
0358	06	980	DB 06H ; ** ;
0359	00	981	DB 00H ; ;
035A	00	982	DB 00H ; ;
		983	
035B	04	984	DB 04H ; * ; [-]
035C	04	985	DB 04H ; * ;
035D	04	986	DB 04H ; * ;
035E	04	987	DB 04H ; * ;
035F	04	988	DB 04H ; * ;
0360	04	989	DB 04H ; * ;
0361	04	990	DB 04H ; * ;
		991	
		992	
0362	00	993	DB 00H ; ; [,]
0363	00	994	DB 00H ; ;
0364	00	995	DB 00H ; ;
0365	01	996	DB 01H ; * ;
0366	00	997	DB 00H ; ;
0367	00	998	DB 00H ; ;
0368	00	999	DB 00H ; ;
		1000	
0369	01	1001	DB 01H ; * ; [/]
036A	02	1002	DB 02H ; * ;
036B	04	1003	DB 04H ; * ;
036C	08	1004	DB 08H ; * ;
036D	10	1005	DB 10H ; * ;
036E	20	1006	DB 20H ; * ;
036F	40	1007	DB 40H ; * ;
		1008	
0370	1D	1009	DB 1DH ; * * * * ; [0]
0371	22	1010	DB 22H ; * * * ;
0372	45	1011	DB 45H ; * * * ;
0373	08	1012	DB 08H ; * * * ;
0374	51	1013	DB 51H ; * * * ;
0375	22	1014	DB 22H ; * * * ;
0376	5C	1015	DB 5CH ; * * * * ;
		1016	
0377	00	1017	DB 00H ; ; [1]
0378	21	1018	DB 21H ; * * ;
0379	40	1019	DB 40H ; * * ;
037A	7F	1020	DB 7FH ; * * * * * ;
037B	00	1021	DB 00H ; ;
037C	01	1022	DB 01H ; * ;
037D	00	1023	DB 00H ; ;
		1024	
037E	23	1025	DB 23H ; * * * ; [2]
037F	44	1026	DB 44H ; * * * ;
0380	01	1027	DB 01H ; * * * ;
0381	48	1028	DB 48H ; * * * ;
0382	01	1029	DB 01H ; * * * ;
0383	48	1030	DB 48H ; * * * ;
0384	31	1031	DB 31H ; * * * ;
		1032	
0385	42	1033	DB 42H ; * * * ; [3]
0386	01	1034	DB 01H ; * * * ;
0387	50	1035	DB 50H ; * * * ;
0388	01	1036	DB 01H ; * * * ;
0389	50	1037	DB 50H ; * * * ;
038A	29	1038	DB 29H ; * * * ;
038B	46	1039	DB 46H ; * * * ;
		1040	
		1041	\$ EJECT

LOC	OBJ	SEQ	SOURCE STATEMENT
038C	04	1042	DB 04H ; * ; [4]
038D	08	1043	DB 08H ; * ;
038E	14	1044	DB 14H ; * * ;
038F	20	1045	DB 20H ; * * * ;
0390	5F	1046	DB 5FH ; * * * * * ;
0391	00	1047	DB 00H ; ;
0392	04	1048	DB 04H ; * ;
		1049	
0393	72	1050	DB 72H ; * * * * ; [5]
0394	01	1051	DB 01H ; * ;
0395	50	1052	DB 50H ; * * ;
0396	01	1053	DB 01H ; * ;
0397	40	1054	DB 40H ; * * ;
0398	11	1055	DB 11H ; * * ;
0399	4E	1056	DB 4EH ; * * * * ;
		1057	
039A	17	1058	DB 17H ; * * * * ; [6]
039B	21	1059	DB 21H ; * * ;
039C	40	1060	DB 40H ; * * * ;
039D	09	1061	DB 09H ; * * * ;
039E	40	1062	DB 40H ; * * * ;
039F	09	1063	DB 09H ; * * * ;
03A0	46	1064	DB 46H ; * * * ;
		1065	
03A1	40	1066	DB 40H ; * ; [7]
03A2	00	1067	DB 00H ; ;
03A3	47	1068	DB 47H ; * * * * ;
03A4	08	1069	DB 08H ; * * ;
03A5	50	1070	DB 50H ; * * * ;
03A6	20	1071	DB 20H ; * * ;
03A7	40	1072	DB 40H ; * * ;
		1073	
03A8	36	1074	DB 36H ; * * * * ; [8]
03A9	49	1075	DB 49H ; * * * * ;
03AA	00	1076	DB 00H ; ;
03AB	49	1077	DB 49H ; * * * * ;
03AC	00	1078	DB 00H ; ;
03AD	49	1079	DB 49H ; * * * * ;
03AE	36	1080	DB 36H ; * * * * ;
		1081	
03AF	30	1082	DB 30H ; * * * * ; [9]
03B0	48	1083	DB 48H ; * * * * ;
03B1	01	1084	DB 01H ; * * * * ;
03B2	48	1085	DB 48H ; * * * * ;
03B3	01	1086	DB 01H ; * * * * ;
03B4	42	1087	DB 42H ; * * * * ;
03B5	3C	1088	DB 3CH ; * * * * ;
		1089	
		1090	
03B6	00	1091	DB 00H ; ; [:]
03B7	00	1092	DB 00H ; ;
03B8	00	1093	DB 00H ; ;
03B9	14	1094	DB 14H ; * * ;
03BA	00	1095	DB 00H ; ;
03BB	00	1096	DB 00H ; ;
03BC	00	1097	DB 00H ; ;
		1098	
03BD	00	1099	DB 00H ; ; [;]
03BE	00	1100	DB 00H ; ;
03BF	01	1101	DB 01H ; * * * * ;
03C0	02	1102	DB 02H ; * * * * ;
03C1	14	1103	DB 14H ; * * * * ;
03C2	00	1104	DB 00H ; ;
03C3	00	1105	DB 00H ; ;
		1106	
03C4	00	1107	DB 00H ; ; [<]
03C5	08	1108	DB 08H ; * * * * ;
03C6	14	1109	DB 14H ; * * * * ;
03C7	22	1110	DB 22H ; * * * * ;
03C8	41	1111	DB 41H ; * * * * ;
03C9	00	1112	DB 00H ; ;
03CA	00	1113	DB 00H ; ;
		1114	
03CB	00	1115	DB 00H ; ; [=]
03CC	14	1116	DB 14H ; * * ;
03CD	00	1117	DB 00H ; ;
03CE	14	1118	DB 14H ; * * ;
03CF	00	1119	DB 00H ; ;
03D0	14	1120	DB 14H ; * * ;
03D1	00	1121	DB 00H ; ;

LOC	OBJ	SEQ	SOURCE STATEMENT
		1122	
03D2	00	1123	DB 00H ; ; [>]
03D3	00	1124	DB 00H ; ;
03D4	41	1125	DB 41H ; ; * *
03D5	22	1126	DB 22H ; ; * *
03D6	14	1127	DB 14H ; ; * *
03D7	08	1128	DB 08H ; ; *
03D8	00	1129	DB 00H ; ;
		1130	
03D9	00	1131	DB 00H ; ; [?]
03DA	20	1132	DB 20H ; ; *
03DB	40	1133	DB 40H ; ; *
03DC	05	1134	DB 05H ; ; * *
03DD	48	1135	DB 48H ; ; * *
03DE	30	1136	DB 30H ; ; **
03DF	00	1137	DB 00H ; ;
		1138	
		1139	END



3065 Bowers Avenue
Santa Clara, California 95051
Tel: (408) 246-7501
TWX: 910-338-0026
TELEX: 34-6372

MICROCOMPUTER AND MEMORY COMPONENT SALES AND MARKETING OFFICES

AUGUST 1977

U.S. AND CANADIAN SALES OFFICES

ALABAMA

Glen White Associates
7844 Horseshoe Trail
Huntsville 35802
Tel: (205) 883-9394

ARIZONA

Sales Engineering, Inc.
7226 Stetson Drive, Suite 34
Scottsdale 85252
Tel: (602) 945-5781
TWX: 910-950-1288
Intel Corp.
8850 N. 35th Avenue
Phoenix 85021
Tel: (602) 242-7205

CALIFORNIA

Intel Corp.*
990 E. Arques Ave.
Suite 112
Sunnyvale 94086
Tel: (408) 738-3870
TWX: 910-339-9279
TWX: 910-338-0255

MAC

P.O. Box 1420
Cupertino 95014
Tel: (408) 257-9880
Earle Associates, Inc.
4805 Mercury Street
Suite L
San Diego 92111
Tel: (714) 278-5441
TWX: 910-335-1585

MAC-I

P.O. Box 8763
Fountain Valley 92708
Tel: (714) 839-3341
Intel Corp.*
1851 East 4th Street
Suite 228
Santa Ana 92701
Tel: (714) 835-9642
TWX: 910-595-1114

COLORADO

Intel Corp.
12075 East 45th Avenue
Suite 310
Denver 80239
Tel: (303) 373-4920
TWX: 910-932-0322

EUROPEAN MARKETING OFFICES

BELGIUM

Intel International*
Rue du Moulin à Papier
51-Boite 1
B-1180 Brussels
Tel: (02) 660 30 10
TELEX: 24814

ORIENT MARKETING OFFICES

JAPAN

Intel Japan Corporation*
Flower Hill-Shinmachi East Bldg.
1-23-9, Shinmachi, Setagaya-ku
Tokyo 154
Tel: (03) 426-9261
TELEX: 781-28426

INTERNATIONAL DISTRIBUTORS

ARGENTINA

S.I.E.S.A.
Av. Pie. Roque Saenz Pena 1142 9B
1035 Buenos Aires
Tel: 35-6784

AUSTRALIA

A. J. Ferguson (Adelaide) PTY, Ltd.
44 Prospect Rd.
Prospect 5082
South Australia 17005
Tel: 269-1244
TELEX: 82635

A. J. Ferguson Electronics

34 Herbert Street
West Ryde, N.S.W. 2114
Tel: AC8 269-1244
TELEX: 82635

Warburton-Frankie Industries

(Melbourne) Pty. Ltd.
199 Parramatta Road
Auburn, N.S.W. 2114
Tel: 648-1711, 648-1381
TELEX: WARFRAN AA 22265

Warburton-Frankie Industries

(Melbourne) Pty. Ltd.
220 Park Street
South Melbourne, Victoria 3205

AUSTRIA

Bacher Elektronische Geräte GmbH
Meidinger Hauptstrasse 78
A 1120 Vienna
Tel: (0222) 83 63 96
TELEX: (01) 1532

BELGIUM

Intel Belgium S.A.
Avenue Val Duchesse, 3
B-1180 Brussels
Tel: (02) 660 00 12
TELEX: 25441

CONNECTICUT

Intel Corp.
Peacock Alley
1 Padanaram Road
Danbury 06810
Tel: (203) 792-8386

FLORIDA

Intel Corp.
2020 W. McNab Road, Suite 104
Ft. Lauderdale 33309
Tel: (305) 971-7200
TWX: 510-956-9407
Intel Corp.
5151 Adanson Street, Suite 105
Orlando 32804
Tel: (305) 628-2393
TWX: 810-853-9219

ILLINOIS

Intel Corp.*
1000 Jorie Boulevard
Suite 224
Oakbrook 60521
Tel: (312) 325-9510
TWX: 910-651-5881

IOWA

Technical Representatives, Inc.
1703 Hillside Drive N.W.
Cedar Rapids 52405
Tel: (319) 396-5662

KANSAS

Technical Representatives, Inc.
801 Clairborne
Olathe 66061
Tel: (913) 782-1177
TWX: 910-749-6412

MARYLAND

Glen White Associates
57 West Timonium Road
Timonium 21093
Tel: (301) 252-6360
Intel Corp.*
57 West Timonium Road
Suite 307
Timonium 21093
Tel: (301) 252-7742
TWX: 710-232-1807

FRANCE

Intel Corporation, S.A.R.L.*
74, Rue D'Arcueil
Sile 223
94528 Rungis Cedex
Tel: (01) 687 22 21
TELEX: 270475

TAIWAN

Taiwan Automation Co.*
6th Floor, 18-1, Lane 14
Chi-Lin Road
Taipei
Tel: (02) 551726-9
TELEX: 11942 TAI AUTO

DENMARK

Scandinavian Semiconductor
Supply A/S
Nannasgade 18
DK-2200 Copenhagen N
Tel: (01) 93 50 90
TELEX: 19037

FINLAND

Oy Fintronic AB
Loennrotinkatu 35D
SF 00180
Helsinki 18
Tel: (90) 664 451
TELEX: 12426

FRANCE

Tekelec Airtronic
Cite des Bruyeres
Rue Carle Vernet
92310 Seures
Tel: (1) 027 75 35
TELEX: 250997

GERMANY

Alfred Neye Enatechnik GmbH
Schillerstrasse 14
D-2085 Quickborn-Hamburg
Tel: (04106) 6121
TELEX: 02-13590

Electronic 2000 Vertriebs GmbH

Neumarkter Strasse 75
D-8000 Muenchen 80
Tel: (089) 434061
TELEX: 522581

Jermyn GmbH

Postfach 1146
D-6277 Kamberg
Tel: (06434) 8005
TELEX: 484426

MASSACHUSETTS

Intel Corp.*
187 Billerica Road, Suite 14A
Chelmsford 01824
Tel: (617) 256-6567
TWX: 710-343-6333

MICHIGAN

Intel Corp.
26500 Northwestern Hwy
Suite 401
Southfield 48075
Tel: (313) 353-0920
TWX: 910-420-1212
TELEX: 2 31143

MINNESOTA

Intel Corp.
8200 Normandale Avenue
Suite 422
Bloomington 55437
Tel: (612) 835-6722
TWX: 910-576-2867

MISSOURI

Technical Representatives, Inc.
Trade Center Bldg.
300 Brookes Drive, Suite 108
Hazelwood 63042
Tel: (314) 731-5200
TWX: 910-762-0618

NEW JERSEY

Intel Corp.
2 Kilmer Road
Edison 08817
Tel: (201) 985-9100
TWX: 710-480-6238

NEW YORK

Intel Corp.*
350 Vanderbilt Motor Pkwy.
Suite 402
Hauppauge 11787
Tel: (516) 231-3300
TWX: 510-221-2198
Intel Corp.
474 Thurston Road
Rochester, N.Y. 14619
Tel: (716) 328-7340
TWX: 519-253-3841

SCANDINAVIA

Intel Scandinavia A/S*
Lyngbyvej 32 2nd Floor
DK-2100 Copenhagen East
Denmark
Tel: (01) 18 20 00
TELEX: 19567
Intel Sweden AB*
Box 20092
S-16120 Bromma
Sweden
Tel: (08) 98 53 90
TELEX: 12261

HONG KONG

ASTEC International
Oriental Centre
14th Floor, No. 67-71
Chatham Road
Kowloon, Hong Kong
Tel: 3-694751
Cable: "ASCOMP"
Telex: 74899 ASCOM HX

INDIA

Electronics International
128 Mahatma Gandhi Road
Secunderabad
Tel: 53211
TELEX: 043-222

ISRAEL

Eastronics Ltd.*
11 Rozanis Street
P.O. Box 38300
Tel-Aviv
Tel: 475151
TELEX: 33638

ITALY

Eletra 35 S.P.A.*
Viale Elvezia, 18
20154 Milan
Tel: (02) 3493041
TELEX: 39332
Eletra 35 S.P.A.*
Via Paolo Gaidano, 141 D
10137 Torino
Tel: (011) 30 97 097 - 30 97 114
Eletra 35 S.P.A.*
Via Giuseppe Valmarana, 63
00139 Rome, Italy
Tel: (06) 81 27 290 - 81 27 324
TELEX: 83051

JAPAN

Pan Electron
No. 1 Higashikata-Machi
Midori-Ku, Yokohama 226
Tel: (045) 471-8811
TELEX: 781-4773

NEW YORK (cont.)

T-Squared
4054 Newcourt Ave
Syracuse 13206
Tel: (315) 463-8592
TWX: 710-541-0554
T-Squared
640 Kream Rd.
P.O. Box W
Pittsford 14534
Tel: (716) 381-2551
TELEX: 97-8289
Intel Corp.
85 Market Street
Poughkeepsie, New York 12601
Tel: (914) 473-2303
TWX: 510-248-0080
NORTH CAROLINA
Glen White Associates
3700 Computer Dr., Suite 330
Raleigh 27609
Tel: (919) 787-7016

OHIO

Intel Corp.*
8312 North Main Street
Dayton 45415
Tel: (513) 880-5350
TELEX: 288-004
Intel Corp.*
26250 Euclid Ave.
Suite 531F
Euclid 44132
Tel: (216) 289-0101

PENNSYLVANIA

Intel Corp.
520 Pennsylvania Ave.
Fort Washington 19034
Tel: (215) 542-9444
TWX: 510-661-0709

TENNESSEE

Glen White Associates
Rt. #12 Noorwood S.D
Jonesboro 37659
Tel: (615) 928-0184
Glen White Associates
2523 Howard Road
Germaniown 38138
Tel: (801) 754-0483

ENGLAND

Intel Corporation (U.K.) Ltd.*
Eroadfield House
4 Between Towns Road
Cowley, Oxford OX4 3NB
Tel: (0865) 77 14 31
TELEX: 837203
Intel Corporation (U.K.) Ltd.
46-50 Beam Street
Nantwich, Cheshire CW5 5LJ
Tel: (0270) 62 65 60
TELEX: 36620

JAPAN (cont.)

Ryoyo Electric Corp.
Konwa Bldg.
1-12-22, Tsukiji, 1-Chome
Chuo-Ku, Tokyo 104
Tel: (03) 543-7711
Nippon Micro Computer Co. Ltd
Mutsumi Bldg. 4-5-21 Kojimachi
Chiyoda-ku, Tokyo 102
Tel: (03) 230-0041

KOREA

Koram Digital
Sam Yung Bldg. #303
71-2 Bukchang - Dong Chung-Ku
Seoul 100

NETHERLANDS

Intel Nederland
AFD Elektronik
Joan Muyskenweg 22
NL-1006 Amsterdam
Tel: (020) 834824
TELEX: 14622

NORWAY

Nordisk Elektronik (Norge) A/S
Mustads Vei 1
N-0502 2
Tel: (02) 55 38 93
TELEX: 16963

PORTUGAL

Dinan
Componentes E. Electronica LDA
Av. Miguel Bombarda, 133
Lisboa 1
Tel: 119 45 313

SOUTH AFRICA

Electronic Building Elements
P.O. Box 4609
Pretoria
Tel: 78 92 21
TELEX: 30181

TEXAS

Mycrosystems Marketing Inc.
13777 N. Central Expressway
Suite 405
Dallas 75231
Tel: (214) 228-7157
TWX: 910-867-4763
Mycrosystems Marketing Inc.
6610 Harwin Avenue, Suite 125
Houston 77036
Tel: (713) 783-2900
Mycrosystems Marketing Inc.
2622 Geronimo Trail
Austin 78746
Tel: (512) 266-1750
Intel Corp.*
2925 L.B.J. Freeway
Suite 100
Dallas 75234
Tel: (214) 241-9521
TWX: 910-860-5487

VIRGINIA

Glen White Associates
P.O. Box 1104
Lynchburg 24505
Tel: (804) 384-6920

WASHINGTON

E.S./Chase Co.
P.O. Box 89903
Seattle 98108
Tel: (206) 762-4824
TWX: 910-444-2298

CANADA

Intel Corp.
70 Chamberlain Ave.
Ottawa, Ontario K1S 1V9
Tel: (613) 232-8576
TELEX: 053-4419
Multitek, Inc.*
4 Barran Street
Ottawa, Ontario K2J 1G2
Tel: (613) 825-4553
TELEX: 053-4585

GERMANY

Intel Semiconductor GmbH*
Seidstrasse 27
8000 Muenchen 2
Tel: (089) 55 61 41
TELEX: 523 177
Intel Semiconductor GmbH
Abraham Lincoln Strasse 30
6200 Wiesbaden 1
Tel: (06121) 74855
TELEX: 04188183
Intel Semiconductor GmbH
D-7000 Stuttgart 80
Ernstthalenstrasse 17
Tel: (0711) 7351506
TELEX: 7255346

Intel Semiconductor GmbH

Wiesenweg 26
D-6272 Niederhausen
Tel: (06127) 2314

SPAIN

Interface
Ronda San Pedro 22
Barcelona 10
Tel: 301 78 51

SWEDEN

Nordisk Elektronik AB
Fack
S-10380 Stockholm 7
Tel: (08) 248340
TELEX: 10547

SWITZERLAND

Industrie AG
Gemsensstrasse 2
Postfach 80 - 21190
CH-8021 Zurich
Tel: (01) 60 22 30
TELEX: 56788

UNITED KINGDOM

Rapid Recall, Ltd.
11-15 Betterton Street
Drury Lane
London WC2H 9BS
Tel: (01) 379-6741
TELEX: 28752
G.E.C. Semiconductors Ltd.
East Lane
Wembley HA9 7PP
Middlesex
Tel: (01) 904-9303
TELEX: 923429
Jermyn Industries
Vestry Estate
Sevenoaks, Kent.
Tel: (0732) 50144
TELEX: 95142

*Field Application Location

